

```

162                                     :***
163                                     : CLEANUP ROUTINES
164                                     :***
165
166                                     :
167                                     : DISCARD 3 BYTES
168
169 000032 062705 000003 POP3:  ADD    #3,R5
170 000036                RETURN
171
172                                     :
173                                     : DISCARD 2 BYTES
174
175 000040 122525 POP2:  CMPB   (R5)+,(R5)+
176 000042                RETURN
177
178                                     :
179                                     : DISCARD 1 BYTE
180
181 000044 005205 POP1:  INC     R5
182 000046                RETURN
183
184                                     :
185                                     : NO CLEANUP
186
187 000050 NULL:  RETURN

```

```

164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179 000000
180 000000 005067 000152'
181 000004 005067 000154'
182 000010 016705 000000G
183 000014 016703 000000G
184 000020 016704 000000G
185 000024 116404 000002
186 000030
187 000034 016702 000160'
188 000040 001421
189 000042 042702 177700
190 000046 016203 000162'
191 000052 016704 000000G
192 000056 026705 000000G
193 000062 001407
194 000064 005002
195 000066 152502
196 000070 001404
197 000072 106302
198 000074
199 000100 000766
200 000102
201
202
203
204
205 000104

;+
; $TMEX5 - EXPAND BINARY INFORMATION INTO REAL INFORMATION
;
; THIS ROUTINE IS CALLED AFTER THE ASCII TEMPLATE FILE HAS BEEN READ
; AND PROCESSED INTO THE BINARY FILE. THE BINARY INFORMATION IS EXPANDED
; AND STORED AT THE LOCATION CONTAINED IN '$ADDR'.
;
; INPUTS:
; $BIN=START OF BINARY INFORMATION
; $PTR=END OF BINARY INFORMATION
; $ADDR=ADDRESS OF WHERE REAL DATA GOES
;
; OUTPUTS:
; ALL REGISTERS=DESTROYED
;
; $TMEX5::
; CLR INDEX ; ZERO INDEX VALUE
; CLR FLAGS ; AND SPECIAL FLAGS
; MOV $BIN,R5 ; START OF BINARY DATA
; MOV $ADDR,R3 ; GET LINE TABLE ADDRESS
; MOV $SLTA,R4 ; GET THE SLT ADDRESS
; MOVB L,DDM(R4),R4 ; GET THE DDM PDV INDEX FOR CALL
; CALL FLT16 ; SEARCH THRU THE UNLOAD BLOCKS
; MOV UBA,R2 ; SUCCESS ?
; BEQ 101$ ; NO
; BIC #^C<77>,R2 ; YES... USE LOW 6 BITS ONLY
; MOV BLOCK(R2),R3 ; GET CONTROLLER TABLE STARTING ADDRESS
; MOV $SLTA,R4 ; RESTORE THE SLT ADDRESS
10$: CMP $PTR,R5 ; IS IT END OF BINARY DATA?
; BEQ 20$ ; IF EQ, YES
; CLR R2 ; GET FUNCTION CODE
; BISB (R5)+,R2 ;
; BEQ 20$ ; IF ZERO, END OF FILE
; ASLB R2 ; CONVERT TO A WORD INDEX
; CALL @TABLE-2(R2) ; EXPAND ONE FIELD
20$: BR 101$ ; KEEP LOOPING UNTIL ENL
; RETURN
;
; ERROR CONDITION
101$: CRASH ; CAN'T FIND LINE TABLE IN UNLOAD BLOCKS

```

771	001554	001411		BEG	10\$; BR IF NO
772	001556	016700	000152'	MOV	INDEX,R0		; GET SECONDARY FILE COUNTER
773	001562	126400	000013	CMPB	L.UNT(R4),R0		; DO UNIT NUMBERS MATCH ?
774	001566	001004		BNE	10\$; NO
775	001570	006300		ASL	R0		; CONVERT TO A WORD INDEX
776	001572	016023	000000G	MOV	\$X2HLD(R0),(R3)+		; STORE HOLDBACK TIMER
777	001576	000401		BR	20\$; AND EXIT
778	001600	005723		10\$:	TST	(R3)+	
779	001602			20\$:	RETURN		
780							
781		000001			.END		

```

241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258 000000
259 000000 005067 000156'
260 000004 005067 000160'
261 000010 005067 000172'
262 000014 005067 000170'
263 000020 016767 000000G 000166'
264 000026 016767 000000G 000174'
265 000034 012700 000000G
266 000040 012701 000000G
267 000044 012702 000176'
268 000050 012122
269 000052 005300
270 000054 003375
271 000056 016767 000000G 000162'
272
273 000064
274
275 000070 016700 000000G
276 000074
277 000100 016704 000000G
278 000104 042704 160000
279 000110 010467 000164'
280 000114 050204
281 000116 056767 000154' 000164'
282 000124 016705 000000G
283 000130 026705 000000G
284 000134 001407
285 000136 005003
286 000140 152503
287 000142 001404
288 000144 106303
289 000146
290 000152 000766
291 000154

;+
; $TMEX6 - EXPAND BINARY INFORMATION INTO REAL INFORMATION
;
; THIS ROUTINE IS CALLED AFTER THE ASCII TEMPLATE FILE HAS BEEN READ
; AND PROCESSED INTO THE BINARY FILE. THE BINARY INFORMATION IS EXPANDED
; AND STORED AT THE LOCATION CONTAINED IN '$ADDR'. THIS ROUTINE IS CALLED
; ONLY FOR LINE TABLES TO BE PLACED AT THE END OF THE PROCESS PARTITION.
;
; INPUTS:
; $BIN=START OF BINARY INFORMATION
; $PTR=END OF BINARY INFORMATION
; $ADDR=ADDRESS OF WHERE REAL DATA GOES
; $BIAS=BIAS OF THE PROCESS PARTITION
;
; OUTPUTS:
; ALL REGISTERS=DESTROYED
;
; $TMEX6::
; CLR INDEX ; ZERO INDEX VALUE
; CLR FLAGS ; AND SPECIAL FLAGS
; CLR NXTLN ; CLEAR COUNTER FOR NUMBER OF .INT'S FOUND
; CLR ICBPT ; CLEAR POINTER INTO ICB
; MOV $ICBAD,ICBAD ; GET START ADDRESS OF ICB
; MOV $ICBLN,ICBLN ; GET ICB LENGTH
; MOV $$SMTS2,R0 ; GET SIZE OF SYMBOL TABLES
; MOV $$PDVA,R1 ; GET START OF TABLE
; MOV #PRPDVA,R2 ; GET START OF LOCAL TABLE
5$: MOV (R1)+(R2)+ ; COPY GLOBAL SYMBOL TABLE
; DEC R0 ; END OF 'APL'
; BGT 5$ ; IF GT, NO
; MOV $FLAGS,PRFLAG ; SAVE STATUS FLAGS
;
; SWSTK$ 20$ ; SWITCH TO SYSTEM STATE
;
; MOV $BIAS,R0 ; GET BIAS OF PROCESS PARTITION
; CALL $MAPX ; MAP TO PARTITION
; MOV $ADDR,R4 ; GET OFFSET INTO PARTITION
; BIC #160000,R4 ; CLEAR THE APR GIVEN
; MOV R4,STLIN ; SAVE START OF LINE TABLE
; BIS R2,R4 ; CLEAR THE APR IN USE
; BIS PRAPR,STLIN ; USE THE APR OF THE PROCESS
; MOV $BIN,R5 ; START OF BINARY DATA
; CMP $PTR,R5 ; IS IT END OF BINARY DATA?
10$: BEQ 20$ ; IF EQ, YES
; CLR R3 ; GET FUNCTION CODE
; BISG (R5)+,R3
; BEQ 20$ ; IF ZERO, END OF FILE
; ASLB 93 ; CONVERT TO A WORD INDEX
; CALL @TABLE-2(R3) ; EXPAND ONE FIELD
; BR 10$ ; KEEP LOOPING UNTIL END
20$: RETURN

```



```

LCCAL DATA
775
776 ; LIBRARY POINTER
777 ;
778 LIBR:
779 MOV (R5)+, (SP) ; ASSEMBLE DESCRIPTOR BIAS
780 MOV (R5)+, 1(SP) ; ...
781 MOV (SP)+, R0 ;
782 MOV (R5)+, -(SP) ; GET ADDRESS OF LIBRARY DESCRIPTOR
783 MOV (R5)+, 1(SP) ; ONE BYTE AT A TIME
784 MOV (SP)+, R3 ; SAVE FOR LATER
785
786 CLR (R4)+ ; ZERO FIRST WORD OF ADDRESS (TEMPORARY)
787 MOV (R5)+, (R4)+ ; ADDRESS OF LIBRARY IS USER DEFINED
788 MOV (R5)+, (R4)+ ; ONE BYTE AT A TIME
789
790 TST -(SP) ; MAKE ROOM TO SAVE AN ARGUMENT
791 MOV (R1), -(SP) ; SAVE CURRENT MAPPING
792 MOV R2, -(SP) ; SAVE BASE ADDRESS ALSO
793
794 MOV R0, (R1) ; MAP TO THE LIBRARY DESCRIPTOR
795 MOV LB.ADR(R3), 4(SP) ; STORE LIBRARY BIAS ON STACK
796 MOV $UBIAS, (R1) ; MAP TO UNLOAD BLOCK
797 MOV $LDA, R2 ; GET LIBRARY DATA ADDRESS WITHIN UNLOAD BLOCK
798 MOV R0, (R2)+ ; STORE DESCRIPTOR BIAS
799 MOV R3, (R2) ; AND OFFSET
800 BIC #^C<77>, (R2)+ ; 6 BITS ONLY
801 MOV R2, $LDA ; UPDATE LIBRARY DATA POINTER
802
803 MOV (SP)+, R2 ; RESTORE OLD BASE ADDRESS
804 MOV (SP)+, (R1) ; RESTORE OLD APR
805 MOV (SP)+, -4(R4) ; SET THE LIBRARY BIAS
806
807 RETURN

```

TMPEX7 - NTL TEMPLATE EXPANSION MACRO V05.03b Saturday 29-Jun-85 ^{B 6} 00:44
Table of contents

5-	53	MACRO DEFINITIONS
11-	721	X3CH
12-	779	X2CH

567	001136			CALL	@TABLE-2(R2)	; EXPAND A FIELD
568	001142	000767		BR	20\$; KEEP LOOKING FOR EOF
569	001144	005267	000350'	INC	INDEX	; TALLY ONE PASS THRU SECONDARY FILE
570	001150	126667	000002 000350'	CMPB	2(SP),INDEX	; ARE WE DONE?
571	001156	001360		BNE	10\$; IF NE, NO
572	001160	022626		CMP	(SP)+,(SP)+	; PURGE STACK
573	001162	005067	000350'	CLR	INDEX	; CLEAR SECONDARY INDEX
574	001166	005067	000352'	CLP	FLAGS	; AND SPECIAL FLAGS
575	001172			RETURN		

TMPEX7 CREATED BY MACRO ON 29-JUN-85 AT 00:44 PAGE 4 B 8

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
\$X2MRT	= ***** GX	6-226
\$X2MWS	= ***** GX	6-216
\$X2RET	= ***** GX	6-231
\$SDCHA	= ***** GX	6-210 6-211
\$SPCHA	= ***** GX	6-212
\$SPRI	= ***** GX	6-213
\$SCSR	= ***** GX	6-206
EXUNL	= ***** GX	10-673
EOF	= ***** GX	8-536 8-564 9-605

TMPINI CREATED BY MACRO ON 29-JUN-85 AT 00:44 PAGE 4 **B 9**

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
CALL	6-131
CRASH	7-244
EMSG\$	#5-57 6-168
EMSG\$R	#5-57 6-185
NKRDF\$	#5-57 5-67
NILR\$	#5-57 5-87 5-88
RETURN	6-176 7-230
SLTDF\$	#5-57 5-59

```

330 .SBTTL .MPTAB,.MXTAB,.MXPTB
331 ;
332 ; ".MPTAB"
333 ;
334 STATES$ MPTAB
335 TRANS$ !FNAME,MPTAB1
336 TRANS$ !ERROR,$EXIT
337 STATES$ MPTAB1
338 TRANS$ !ONLY1,$EXIT,$MPTB
339 ;
340 ; ".MXTAB"
341 ;
342 STATES$ MXTAB
343 TRANS$ !FNAME,MXTAB1
344 TRANS$ !ERROR,$EXIT
345 STATES$ MXTAB1
346 TRANS$ !ONLY1,$EXIT,$MXTAB
347 ;
348 ; ".MXPTB"
349 ;
350 STATES$ MXPTB
351 TRANS$ !FNAME,MXPTB1
352 TRANS$ !ERROR,$EXIT
353 STATES$ MXPTB1
354 TRANS$ !ONLY1,$EXIT,$MXPTB
355 ;
356 ;
357 ;

```

```

115 ;+ $TMVCT - SETUP INTERRUPT VECTOR(S)
116 ;
117 ; INPUTS:
118 ; $SVECT=VECTOR TO BE USED
119 ; $NVECT=NUMBER OF VECTORS TO SET UP
120 ; $ADDR=LINE TABLE ADDRESS
121 ; $VECT1=INTERRUPT DISPLACEMENT VALUE(S)
122 ; $SLTA=SLT ADDRESS
123 ; $ICBAD=ADDRESS OF ICB
124 ;
125 ; OUTPUTS:
126 ; ALL REGISTERS=DESTROYED
127 ;
128 ;-
129 $TMVCT::
130 000000 TST $FLAG1 ; ANY ERRORS YET?
131 000000 BMI 2$ ; IF M1, YES - DON'T CHECK
132 000004 100430 MOV $PDVA,R1 ; GET PDV ADDRESS
133 000012 032761 000000G 000000G BIT #ZF.PSE,Z.FLG(R1) ; IS THIS A PSEUDO DEVICE?
134 000020 001022 BNE 2$ ; IF NE, YES - DON'T CHECK
135 000022 032767 000000G 000000G BIT #FL.DDM,$FLAGS ; ARE WE LOADING A DDM?
136 000030 001416 BEQ 2$ ; IF EQ, NO - DON'T CHECK
137 000032 032767 000000G 000000G BIT #FL.CHA,$FLAGS ; MUX CHARACTERISTICS UPDATE ONLY ?
138 000040 001012 BNE 2$ ; YES - DON'T CHECK
139 000042 016700 MOV $SVECT,R0 ; GET FIRST VECTOR ADDRESS
140 000046 012701 000000G MOV $SVECT1,R1 ; GET VECTOR DISPLACEMENT(S) ADDRESS
141 000052 016702 000000G MOV $NVECT,R2 ; GET NUMBER OF VECTORS
142 000056 010203 MOV R2,R3 ; TWICE
143 000060 060301 ADD R3,R1 ; POINT 1 WORD PAST LAST DISPLACEMENT
144 000062 060301 ADD R3,R1 ;
145 000064 000402 BR 3$ ; CONTINUE
146
147 000066 2$: CALLR 60$ ; CONTINUE
148 000072 3$:
149
150 .IF DF R$11M
151 000072 SWSTK$ 50$ ;* ENTER KERNEL MODE
152 .IF DF R$MPL
153 MOV @KSAR5,-(SP) ;* SAVE CURRENT MAPPING
154
155 .IFF
156
157 000076 010005 4$: MOV R0,R5 ;* COPY VECTOR ADDRESS
158 000100 042705 000077 BIC #77,R5 ;* MAKE IT A MULTIPLE OF 100
159 000104 ASR$ 5,R5 ;* CONVERT TO A WORD INDEX
160 ;* BY SHIFTING RIGHT 5 PLACES
161 000116 016504 000000' MOV NS1TAB(R5),R4 ;* GET VECTOR CONTENTS
162 000122 022014 CMP (R0)+,(R4) ;* IS VECTOR STILL AVAILABLE?
163 000124 001034 BNE 30$ ;* IF NE, NO
164 000126 005720 TST (R0)+ ;* SKIP VECTOR PRIORITY
165 000130 SOB R2,4$ ;* LOOP $NVECT TIMES
166
167 .IFT
168
169 MOV $SLTA,R5 ;* GET SLT ADDRESS
170 MOV L.KRBA(R5),R5 ;* TO GET THE KRB ADDRESS
171 MOV$ @NCPU,R4 ;* GET THE NUMBER OF PROCESSORS

```

FILEID**UNLHC

```

UU      UU      NN      NN      LL      HH      HH      CCCCCCCC
UU      UU      NN      NN      LL      HH      HH      CCCCCCCC
UU      UU      NN      NN      LL      HH      HH      CC
UU      UU      NN      NN      LL      HH      HH      CC
UU      UU      NNNN     NN      LL      HH      HH      CC
UU      UU      NNNN     NN      LL      HH      HH      CC
UU      UU      NN      NN      LL      HHHHHHHHHH    CC
UU      UU      NN      NN      LL      HHHHHHHHHH    CC
UU      UU      NN      NNNN     LL      HH      HH      CC
UU      UU      NN      NNNN     LL      HH      HH      CC
UU      UU      NN      NN      LL      HH      HH      CC
UU      UU      NN      NN      LL      HH      HH      CC
UUUUUUUUUU      NN      NN      LLLLLLLLLL    HH      HH      CCCCCCCC
UUUUUUUUUU      NN      NN      LLLLLLLLLL    HH      HH      CCCCCCCC

```

```

....
....
....
....

```

```

LL      SSSSSSSS  TTTTTTTTTT
LL      SSSSSSSS  TTTTTTTTTT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LL      SSSSSS    TT
LL      SSSSSS    TT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LLLLLLLLLL  SSSSSSSS  TT
LLLLLLLLLL  SSSSSSSS  TT

```



```

52
53
54
55
56
57
58
59
60 000000
61 000000
62 000000
63 000000
64 000000
65 000000
66 000000
67 000000
68 000000
69 000000
70 000000
71 000000
72
73
74
75
76
77
78
79
80
81

      .SBTTL  MACRO DEFINITIONS

      ***
      ; LIBRARY MACROS
      ***
      .MCALL  PVCDF$,CUGDF$,RDTDF$,DSTDF$,DTEDF$,XCBDF$,PHBDF$
      .MCALL  CHNDF$,SVCDF$,X29DF$,TRADF$,ASL$,SAVRG,RESRG,TCBDF$

      CHNDF$      ; DEFINE OUTGOING CHANNELS LIST OFFSETS
      CUGDF$      ; DEFINE CLOSED USER GROUP BLOCK OFFSETS
      DSTDF$      ; DEFINE X.25/X.29 DESTINATION BLOCK OFFSETS
      DTEDF$      ; DEFINE LOCAL DTE DESCRIPTOR OFFSETS
      PHBDF$      ; DEFINE X.25 HOME BLOCK OFFSETS
      PVCDF$      ; DEFINE PVC OFFSETS
      RDTDF$      ; DEFINE REMOTE DTE BLOCK OFFSETS
      SVCDF$      ; DEFINE SVC BLOCK OFFSETS
      XCBDF$      ; DEFINE X.25 CIRCUIT BLOCK OFFSETS
      X29DF$      ; DEFINE X.29 DATA BLOCK OFFSETS
      TCBDF$      ; DEFINE TCB DATA STRUCTURE OFFSETS
      TRADF$      ; DEFINE TRACE DATA STRUCTURE OFFSETS

      ; LOCAL MACRO DEFINITIONS
      ;
      .MACRO  SAVMAP
      MOV     @KSAR5,-(SP)      ; SAVE APR 5
      .ENDM

      .MACRO  RESMAP
      MOV     (SP)+,@KSAR5      ; RESTORE APR5
      .ENDM

```

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

ASLS	#5-58	9-225	12-350						
CALL	7-115	7-118	7-121	7-124	7-126	7-128	7-129	7-130	7-131
	7-135	8-154	8-162	8-177	9-201	9-208	9-219	9-235	9-241
	10-270	10-281	10-289	11-315	12-336	12-362			7-132
CHNDFS	#5-58	5-60							10-267
CUGDFS	#5-57	5-61							
DSTDFS	#5-57	5-62							
DTEDFS	#5-57	5-63							
PHBDFS	#5-57	5-64							
PVCDFS	#5-57	5-65							
RDIDFS	#5-57	5-66							
RESMAP	#5-79	8-180	9-244	10-283	10-290				
RESRG	#5-58	9-239	9-240	10-282	12-360	12-361			
RETURN	7-137	8-181	9-245	10-291	11-316	12-363			
SAVMAP	#5-75	8-155	9-202	10-268	10-279				
SAVRG	#5-58	9-212	9-213	10-280	12-346	12-347			
SVCDFS	#5-58	5-67							
SWSTKS	8-154	9-201	10-267	12-336					
TCBDFS	#5-58	5-70							
TRADFS	#5-58	5-71							
XCBDFS	#5-57	5-68							
X29DFS	#5-58	5-69							

UNLSUB - NTL UNLOAD SUBROUTINES MACRO V05.03b Saturday 29-Jun-85 00:48 Page 12-1
Symbol table

A\$\$CHK= 000000	K\$\$LDC= 000000	LN.ON = 000000	L.UNT 000013	SF.LPB= 000004
A\$\$CPS= 000000	K\$\$TPS= 000074	LN.OOP= 000004	M\$\$CRB= 000124	SF.MFL= 000040
A\$\$PRI= 000000	K.CSR 000002	LN.OPE= 000001	M\$\$CRX= 000000	SF.PAC= 000020
A\$\$TRP= 000000	K.PRI 000000	LN.REF= 000002	M\$\$FCS= 000000	SF.REA= 000010
C\$\$CKP= 000000	K.VCT 000001	LN.SER= 000002	M\$\$MGE= 000000	SF.SER= 000001
C\$\$ORE= 000400	LD\$LP = 000000	LN.STA= 000017	M\$\$NET= 000000	SF.SVC= 000002
C\$\$RSH= 177546	LF.ACT= 100000	LN.SUB= 000360	M\$\$OVR= 000000	SF.UNL= 000040
DDMAC = ***** GX	LF.BRO= 000400	LN.TRI= 000006	N\$\$ACC= 000001	SLTMA = ***** GX
DLCAC = ***** GX	LF.BWT= 000007	LS\$ASG= 000000	N\$\$BUF= 000001	SLTNM = ***** GX
D\$\$BUG= 177514	LF.ENA= 002000	LS\$DRV= 000000	N\$\$LDV= 000001	S\$\$WRG= 000000
D\$\$ISK= 000000	LF.LPB= 001000	LS\$PT1= 000001	N\$\$MCP= 000001	S\$\$YSZ= 007600
D\$\$L11= 000001	LF.MDC= 000100	LS\$11R= 000000	N\$\$MLL= 000001	S.COST 000001
D\$\$YNC= 000000	LF.MFL= 004000	L.COST 000015	N\$\$MOV= 000010	S.FLG 000000
D\$\$YNM= 000000	LF.MTP= 000020	L.CTL 000012	N\$\$NCT= 000001	S.LEN 000004
E\$\$XPR= 000000	LF.PAC= 000200	L.CVA 177776	N\$\$PEM= 000001	S.NMST 000002
FNDADD 000074RG	LF.RDY= 040000	L.DDM 000002	PDVNM = ***** GX	S.OWNR 000003
F\$\$VL= 000001	LF.REA= 010000	L.DDS 000004	PDVTA = ***** GX	S.OWNR 000000
GETIND 000002RG	LF.SER= 000040	L.DLC 000003	P\$\$P45= 000000	T\$MIN= 000000
GETPDV 000044RG	LF.TIM= 000.10	L.DLM 000006	P\$\$WRD= 000000	UNITS 000000R
G\$\$TPP= 000000	LF.UNL= 020000	L.DLS 000010	Q\$OPT= 000010	V\$CTR= 001000
G\$\$TSS= 000000	LF.X2P= 000000	L.FLG 000000	R\$DER= 000000	XKRB 000056RG
G\$\$TIK= 000000	LN.CLO= 000000	L.KRBA 000016	R\$K11= 000001	X\$DBI= 000000
G\$\$WRD= 000000	LN.DUM= 000005	L.LEN = 000022	R\$SND= 000000	\$DECEX= ***** GX
I\$RAR= 000000	LN.LOA= 000004	L.MPF 000022	R\$11M= 000000	\$ICBLN= ***** GX
I\$RDN= 000000	LN.LOO= 000003	L.NMST 000020	SCAN 000136RG	\$KRBLN= ***** GX
K\$CNT= 177546	LN.OAU= 000003	L.NSTA 000014	SF.ACT= 000200	.CBIAS= ***** GX
K\$CSR= 177546	LN.OFF= 000001	L.OWNR 000021	SF.ENA= 000100	

. ABS. 177776 000 (RW,I,GBL,ABS,OVR)
000252 001 (RW,I,LCL,REL,CON)
Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 10850 Words (43 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:10.08
SY:UNLSUB.V2,[132,134]UNLSUB/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]UNLSUB

.TITLE XPAR - RELEASE PARTITION CONTROL BLOCK
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

NTL - PARTITION CONTROL BLOCK DE-ALLOCARION ROUTINE

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/Rsx V1.0

```

189
190
191
192
193
194
195
196 000052 112546
197 000054 112566 000001
198 000060 012600
199 000062 112546
200 000064 112566 000001
201 000070 012601
202 000072 001436
203
204
205
206 000074 022700 120000
207 000100 101031
208 000102 016704 000000G
209 000106
210 000112 010003
211 000114 010105
212 000116 016400 000000G
213 000122
214 000126 010300
215 000130 010501
216 000132 012703 177776G
217 000136 062704 000002G
218 000142 011463 000002
219 000146
220 000152 016714 000000G
221 000156 042714 160000
222 000162 000402
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245

```

```

***
DE-ALLOCATION CLEANUP ROUTINES
***

; SINGLEWORD ADDRESSABLE STORAGE
XSOM:  MOVB (R5)+,(SP) ; ASSEMBLE ADDRESS
        MOVB (R5)+,1(SP) ; ...
        MOV (SP)+,R0 ; ...
        MOVB (R5)+,(SP) ; AND COUNT
        MOVB (R5)+,1(SP) ; ...
        MOV (SP)+,R1 ; ...
        BEQ 10$ ; IF ZERO, DO NOTHING

        .IF DF R$$$11M
        .IF DF M$$$MGE
        CMP #120000,R0 ; IS ALLOCATION FROM PARTITION SPACE ?
        BHI 5$ ; NO
        MOV $PCB,R4 ; GET PCB ADDRESS
        SWSTK$ 10$ ; * ENTER KERNEL MODE
        MOV R0,R3 ; * SAVE ALLOCATION ADDRESS
        MOV R1,R5 ; * AND SIZE
        MOV P.REL(R4),R0 ; * MAP TO THE PROCESS
        CALL $MAPX ; *
        MOV R3,R0 ; * RESTORE ALLOCATION ADDRESS
        MOV R5,R1 ; * AND SIZE
        MOV #SPRAVL-2,R3 ; * POINT AT COMMON LISTHEAD
        ADD #P.WAIT+2,R4 ; * POINT AT PCB ALLOCATION WORD
        MOV (R4),2(R3) ; * SETUP PARTITION FREE SPACE POINTER
        CALL $DEAC2 ; * DE-ALLOCATE BACK TO PARTITION SPACE
        MOV $SPRAVL,(R4) ; * RESTORE PCB ALLOCATION WORD
        BIC #160000,(R4) ; ...
        BR 10$ ; *

        .IFF
        MOV $PCB,R4 ; GET PCB ADDRESS
        MOV P.REL(R4),R3 ; AND PARTITION STARTING ADDRESS
        CMP R0,R3 ; IS ALLOCATION FROM PARTITION SPACE ?
        BLO 5$ ; NO
        ADD P.SIZE(R4),R3 ; GET PARTITION ENDING ADDRESS
        CMP R0,R3 ; IS ALLOCATION FROM PARTITION SPACE ?
        BHS 5$ ; NO
        MOV #SPRAVL-2,R3 ; POINT AT COMMON LISTHEAD
        ADD #P.WAIT+2,R4 ; POINT AT PCB ALLOCATION WORD
        MOV (R4),2(R3) ; SETUP PARTITION FREE SPACE POINTER
        CALL $DEAC1 ; DE-ALLOCATE BACK TO PARTITION SPACE
        MOV $SPRAVL,(R4) ; RESTORE PCB ALLOCATION WORD
        BR 10$

        .ENDC
        .ENDC

5$: CALL $DEA16 ; DEALLOCATE SINGLEWORD STORAGE
10$: RETURN

; DOUBLEWORD ADDRESSABLE STORAGE
;

```

```

207 ***
208 EXPANSION ROUTINES
209 WITH THE EXCEPTION OF ".DVCHA" AND ".SECSR", ALL PROCESSING CONSISTS
210 OF UPDATING THE BINARY BUFFER POINTER (R5) AND THE LINE TABLE POINTER
211 (R3).
212 ***
213
214 ; LOCAL PHYSICAL STATION ADDRESS (BYTE)
215 ADDR: INC R3
216 RETURN
217
218 ; LOCAL PHYSICAL STATION ADDRESS (WORD)
219 ADDR: TST (R3)+
220 RETURN
221
222 ; ZERO BYTE(S)
223 BLKW: CLR R0 ; GET REPEAT COUNT
224 BISB (R5)+,R0 ;
225 BNE 10$ ; ZERO BECOMES 256.
226 MOV #256.,R0 ;
227 10$: ADD R0,R3
228 RETURN
229
230 ; ZERO WORD(S)
231 BLKW: CLR R0 ; GET REPEAT COUNT
232 BISB (R5)+,R0 ;
233 BNE 10$ ; ZERO BECOMES 256.
234 MOV #256.,R0 ; 2 BYTES PER WORD
235 ASL R0
236 10$: ADD R0,R3
237 RETURN
238
239 ; SINGLE BYTE
240 BYTE: CMPB (R5)+,(R3)+
241 RETURN
242
243 ; POINTER TO ALLOCATED CORE
244 CORE: .IF DF M$MGE
245 ADD #6,R5
246 CMP (R3)+,(R3)+
247 .IFF
248
249
250
251
252
253 000154 122523
254 000156
255
256
257
258
259 000160
260
261 000160 062705 000006
262 000164 022323
263

```

ADDRB 000106R	K\$CNT= 177546	LSTHD 000330R	N\$PEM= 000001	UNB 000506R
ADDRW 000112R	K\$CSR= 177546	LTAB 000334R	PECHA 000252R	UNW 000512R
APR 001070R	K\$LDC= 000000	L\$ASG= 000000	PRI 000340R	V\$CTR= 001000
A\$CHK= 000000	K\$TPS= 000074	L\$DRV= 000000	PRIOV 000376R	WORD 000516R
A\$CPS= 000000	K.CSR 000002	L\$P11= 000001	PR7 = ***** GX	X\$DBT= 000000
A\$PRI= 000000	K.VCT 000001	L\$11R= 000000	P\$P45= C00000	X2CH01 001370R
A\$TRP= 000000	LD\$LP = 000000	L.COST 000015	P\$WRD= 000000	X2CH02 001424R
BLKB 000116R	LF.LF 000640R	L.CTL 000012	Q\$OPT= 000010	X2CH03 001460R
BLKW 000134R	002 LF.ACT= 100000	L.CVA 177776	R\$DER= 000000	X2CH04 001514R
BLOCK 000162R	LF.BRO= 000400	L.DDM 000002	R\$K11= 000001	X2CH05 001550R
BYTE 000154R	LF.BWT= 000007	L.DDS 000004	R\$SND= 000000	X3CH01 001314R
CORE 000160R	LF.ENA= 002000	L.DLC 000003	R\$11M= C00000	X3CH02 001320R
CSR 000170R	LF.LPB= 001000	L.DLM 000006	SCDM 000404R	X3CH03 001324R
CTIM 000176R	LF.MDC= 000100	L.DLS 000010	SECSR 000414R	X3CH04 001330R
C\$CKP= 000000	LF.MFL= 004000	L.FLG 000000	SF.ACT= 000200	X3CH05 001334R
C\$ORE= 000400	LF.MTP= 000020	L.KRBA 000016	SF.ENA= 000100	X3CH06 001340R
C\$RSH= 177564	LF.PAC= 000200	L.LEN = 000022	SF.LPB= 000004	X3CH07 001344R
DPRB 001074R	LF.RDY= 040000	L.MPF 000022	SF.MFL= 000040	X3CH08 001350R
DPRW 001100R	LF.REA= 010000	L.NMST 000020	SF.PAC= 000020	X3CH09 001354R
DVCHA 000202R	LF.SER= 000040	L.NSTA 000014	SF.REA= 000010	X3CH10 001360R
D\$BUG= 177514	LF.TIM= 000010	L.OWNR 000021	SF.SER= 000001	X3CH11 001364R
D\$ISK= 000000	LF.UNL= 020000	L.UNT 000013	SF.SVC= 000002	ZTIME = ***** GX
D\$L11= 000001	LF.X2P= 000000	MPTAB 000636R	SF.UNL= 000040	\$ADDR = ***** GX
D\$YNC= 000000	LIBR 000314R	MXPTB 000722R	SLNB 000466R	\$BIN = ***** GX
D\$YNM= 000000	LINKS 000324R	MXTAB 000524R	SLNW 000472R	\$FLAGS= ***** GX
EOF 000302R	LN.CLC= 000000	M\$CRB= 000124	STNB 000476R	\$MAPX = ***** GX
E\$XPR= 000000	LN.DUM= 000005	M\$CRX= 000000	STNW 000502R	\$PTR = ***** GX
FILE 000530R	LN.LCA= 000004	M\$FCS= 000000	S\$WRG= 000000	\$SLTA = ***** GX
FLAGS 000154R	LN.LOD= 000003	M\$HGE= 000000	S\$YSZ= 007600	\$TMEX5 000000R6
FLT 001104R	LN.OAU= 000003	M\$NET= 000000	S.COST 000001	\$X2HLD= ***** GX
FLT16 001104R	LN.OFF= 000001	M\$OVR= 000000	S.FLG 000000	\$X2MBS= ***** GX
FL.KMX= ***** GX	LN.ON= 000000	NOD 000372R	S.LEN 000004	\$X2MRT= ***** GX
F\$LVL= 000001	LN.OOP= 000004	NTLPT = ***** GX	S.NMST 000002	\$X2MWS= ***** GX
G\$TPP= 000000	LN.OPE= 000001	N\$ACC= 000001	S.OWNR 000003	\$X2RET= ***** GX
G\$TSS= 000000	LN.REF= 000002	N\$BUF= 0.0001	TABLE 000000R	\$SDCHA= ***** GX
G\$TTK= 000000	LN.SER= 000002	N\$LDV= 000001	TIME 001042R	\$SPCHA= ***** GX
G\$WRD= 000000	LN.STA= 000017	N\$MCP= 000001	T\$KMG= 000000	\$SPRI = ***** GX
INDEX 000152R	LN.SUS= 000360	N\$MLI= 000000	T\$MIN= 000000	\$SSCR= ***** GX
INT 000304R	LN.TRI= 000006	N\$MOV= 000010	UBA 000160R	002 .CXUNL= ***** GX
I\$RAR= 000000		N\$NCT= 000001	UBIAS 000156R	002 .EOF = ***** GX
I\$RDN= 000000				

. ABS. 177776 000 (RW,I,GBL,ABS,OVR)
001604 001 (RW,I,LCL,REL,CON)
DATA 000262 002 (RW,D,LCL,REL,CON)
Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 10930 Words (43 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:17.19
SY:TMPEX5.V2,L132,134]TMPEX5/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,1:0]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]TMPEX5

```

293      ;**
294      ; EXPANSION ROUTINES
295      ;**
296
297      ; LOCAL PHYSICAL STATION ADDRESS (BYTE)
298
299      300 000156 032767 000000C 000162' ADDR: BIT    #FL.DDM!FL.DLC,PRFLAG ; IS IT A DDM?
301      301 000164 001403          BEQ    10$          ; BR IF NO
302      302 000165 116724 001224'          MOVB   PRLSAD,(R4)+ ; STORE LOCAL STATION ADDRESS
303      303 000172          RETURN
304      304 000174 112724 177777          10$: MOVB   #-1,(R4)+ ; LSAD NOT VALID FOR LLC
305      305 000200          RETURN
306
307      ; LOCAL PHYSICAL STATION ADDRESS (WORD)
308
309      310 000202 032767 000000C 000162' ADDR: BIT    #FL.DDM!FL.DLC,PRFLAG ; IS IT A DDM?
311      311 000210 021405          BEQ    10$          ; BR IF NO
312      312 000212 005003          CLR     R3          ; GET LOCAL STATION ADDRESS
313      313 000214 156703 001224'          BLSB   PRLSAD,R3 ;
314      314 000220 010324          MOV     R3,(R4)+ ; STORE IT
315      315 000222          RETURN
316      316 000224 012724 177777          10$: MOV     #-1,(R4)+ ; LSAD NOT VALID FOR LLC
317      317 000230          RETURN
318
319      ; ZERO BYTE(S)
320
321      321 000232 112500          BLKB:  MOVB   (R5)+,R0 ; GET BYTE COUNT
322      322 000234 105024          10$:  CLRB   (R4)+ ; STORE A ZERO BYTE
323      323 000236 105300          DECIB  R0 ; LOOP BETWEEN 1 AND 256. TIMES
324      324 000240 001375          BNE     10$
325      325 000242          RETURN
326
327      ; ZERO WORD(S)
328
329      330 000244 112500          BLKW:  MOVB   (R5)+,R0 ; GET WORD COUNT
331      331 000246 005024          10$:  CLR    (R4)+ ; STORE A ZERO WORD
332      332 000250 105300          DECIB  R0 ; LOOP BETWEEN 1 AND 256. TIMES
333      333 000252 001375          BNE     10$
334      334 000254          RETURN
335
336      ; SINGLE BYTE
337
338      339 000256 112524          BYTE:  MOVB   (R5)+,(R4)+ ; COPY A BYTE
340      340 000260          RETURN
341
342      ; CSR
343
344      345 000262 112500          CSR:   MOVB   (R5)+,R0 ; GET MODIFICATION VALUE
346      346 000264 065700 000342'          ADD    PRCSR,R0 ; ADD IN CSR ADDRESS
347      347 000270 010024          MOV     R0,(R4)+ ; STORE THE RESULT
348      348 000272          RETURN
349

```



```

809                                     ; SECONDARY LINKED FILE
810
811
812 001754 012767 100000 000160' MXPTB: MOV #100000,FLAGS ; INDICATE MUX EXPANSION
813 001762 010246 MOV R2,-(SP) ; SAVE CURRENT APR SELECTION BITS
814 001764 016746 000154' MOV PRAPR,-(SP) ; AND CURRENT PROCESS APR SELECTION
815 001770 005002 CLR R2 ; NO RELOCATION NEEDED
816 001772 005067 000154' CLR PRAPR
817 001776 112546 MOV (R5)+,-(SP) ; GET REPEAT COUNT
818 002000 112524 MOV (R5)+,(R4)+ ; GET MUX TABLE ADDRESS
819 002002 112524 MOV (R5)+,(R4)+
820 002004 010446 MOV R4,-(SP) ; SAVE OUTPUT FILE POINTER
821 002006 016404 177776 MOV -2(R4),R4 ; SET NEW OUTPUT FILE POINTER
822 002012 010546 MOV R5,-(SP) ; SAVE CURRENT BINARY FILE POINTER
823 002014 011605 10$: MOV (SP),R5 ; RESTORE BINARY FILE POINTER
824 002016 005003 20$: CLR R3 ; GET FUNCTION CODE
825 002020 152503 BIS (R5)+,R3
826 002022 122703 000000G CMPB #EOP,R3 ; IS IT END OF FILE ?
827 002026 001404 BEQ 30$ ; IF EQ, YES
828 002030 106303 ASLB R3 ; CONVERT TO A WORD INDEX
829 002032 CALL @TABLE-2(R3) ; EXPAND A FIELD
830 002036 000767 BR 20$ ; GO TILL END OF FILE
831 002040 005267 000156' 30$: INC INDEX ; TALLY ONE PASS THRU SECONDARY FILE
832 002044 126667 000004 000156' 000004 CMPB 4(SP),INDEX ; ARE WE DONE ?
833 002052 001360 BNE 10$ ; IF NE, NO
834 002054 005726 TST (SP)+ ; PURGE BINARY FILE POINTER
835 002056 012604 MOV (SP)+,R4 ; RESTORE OUTPUT FILE POINTER
836 002060 005726 TST (SP)+ ; PURGE COUNT
837 002062 012667 000154' MOV (SP)+,PRAPR ; RESTORE APR SELECTION BITS
838 002066 012602 MOV (SP)+,R2
839 002070 005067 000156' CLR INDEX ; CLEAR SECONDARY INDEX
840 002074 005067 000160' CLR FLAGS ; AND SPECIAL FLAGS
841 002100 016703 000000G MOV $MXSIZ,R3 ; GET SIZE OF MUX TABLES
842 002104 CALLR SAVALL ; SAVE SIZE IN UNLOAD BLOCK
843
844
845                                     ; CURRENT TIME
846
847
848 002110 016767 000000G 000000' TIME: MOV ZTIME,TEMP ; STORE $ZTIME ADDRESS
849 002116 062767 000002 000000' ADD #2,TEMP ; GET $ZTIME+2 ADDRESS
850 002124 117724 000000' MOV @TEMP,(R4)+ ; STORE TIME LOW BYTE
851 002130 005267 000000' INC TEMP ; GET $ZTIME+3 ADDRESS
852 002134 117724 000000' MOV @TEMP,(R4)+ ; STORE TIME HIGH BYTE
853 002140 RETURN
854
855                                     ; ACTIVE POLLING RATIO (APR)
856
857
858
859 002142 105767 000160' APR: TSTB FLAGS ; IS IT MTP EXPANSION ?
860 002146 100005 BPL 101$ ; IF PL, NO
861 002150 016700 000156' MOV INDEX,R0 ; GET CURRENT STATION INDEX
862 002154 116024 000432' MOV PRAPR(R0),(R4)+ ; STORE APR
863 002160 RETURN
864 002162 112724 177777 101$: MOV #-1,(R4)+ ; APR NOT IN MPTAB FILE
865 002166 RETURN

```

.TITLE TMPEX7 - NTL TEMPLATE EXPANSION ROUTINES (UPDATE)
.IDENT /V05.00/

.IIF NDF M\$\$MGE, .END

COPYRIGHT (C) 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

NTL - MUX CHARACTERISTICS UPDATE EXPANSION ROUTINES FOR PART. LINE TAB.

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 15-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/R SX V1.0

```

577 ; SECONDARY LINKED FILE
578
579
580 001174 005267 000352' MXPTB: INC FLAGS ; INDICATE MUX EXPANSION
581 001200 005723 TST (R3)+ ; POINT PAST FIRST LINE TABLE ADDRESS
582 001202 010346 MOV R3,-(SP) ; SAVE OUTPUT FILE POINTER
583 001204 112546 MOVB (R5)+,(SP) ; GET REPEAT COUNT
584 001206 122525 CMPB (R5)+,(R5)+ ; SKIP OVER ADDRESS
585 001210 016303 MOV -2(R3),R3 ; GET MUX TABLE ADDRESS
586 001214 010546 MOV R5,-(SP) ; SAVE CURRENT BINARY FILE POINTER
587 001216 011605 MOV (SP),R5 ; RESTORE BINARY FILE POINTER
588 001220 126467 000013 000350' 10$: CMPB L,UNT(R4),INDEX ; DO UNIT NUMBERS MATCH ?
589 001226 001003 BNE 20$ ; IF NE, NO
590
591 .IF DF R$$$11D ! I$$$AS
592 MOV R3,-(SP) ; GET LINE TABLE ADDRESS
593 BIC #160000,(SP) ; CLEAR MAPPING BITS
594 BIS #60000,(SP) ; SLT LINE TABLE ADDRESS USES APR3
595 CMP L,DDS(R4),(SP)+ ; IS THE LINE TABLE POINTER CORRECT ?
596
597 .IFF
598
599 001230 026403 0000C4 CMP L,DDS(R4),R3 ; IS THE LINE TABLE POINTER CORRECT ?
600 .ENDC
601
602 001234 001026 20$: BNE 101$ ; IF NE, NO .. ERROR
603 001236 005002 CLR R2 ; GET FUNCTION CODE
604 001240 152502 BISB (R5)+,R2 ;
605 001242 122702 CMPB #.EOF,R2 ; IS IT END OF FILE ?
606 001246 001404 BEQ 30$ ; IF EQ, YES
607 001250 106302 ASLB R2 ; CONVERT TO A WORD INDEX
608 001252 CALL @TABLE-2(R2) ; EXPAND A FIELD
609 001256 000767 BR 20$ ; GO TILL END OF FILE
610 001260 005267 000350' 30$: INC INDEX ; TALLY ONE PASS THRU SECONDARY FILE
611 001264 126667 000002 000350' CMPB 2(SP),INDEX ; ARE WE DONE ?
612 001272 001351 BNE 10$ ; IF NE, NO
613 001274 022626 CMP (SP)+,(SP)+ ; PURGE STACK
614 001276 012603 MOV (SP)+,R3 ; RESTORE OUTPUT FILE POINTER
615 001300 005067 000350' CLR INDEX ; CLEAR SECONDARY INDEX
616 001304 005067 000352' CLR FLAGS ; AND SPECIAL FLAGS
617 001310 RETURN
618
619 ; ERROR CONDITIONS
620
621 001312 101$: CRASH ; LINE TABLE POINTER (R3) IS SCREWED UP
622
623 ; CURRENT TIME
624
625
626
627 001314 016767 000000G 000000' TIME: MOV ZTIME,TEMP ; STORE $ZTIME ADDRESS
628 001312 062767 000002 000000' ADD #2,TEMP ; GET $ZTIME+2 ADDRESS
629 001330 117723 000000' MOVB @TEMP,(R3)+ ; STORE TIME LOW BYTE
630 001334 005267 000000' INC TEMP ; GET $ZTIME+3 ADDRESS
631 001340 117723 000000' MOVB @TEMP,(R3)+ ; STORE TIME LOW BYTE
632 001344
633

```

TMPEX7 CREATED BY MACRO ON 29-JUN-85 AT 00:44 PAGE 5 C 8

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES									
CALL	6-242	6-249	6-253	6-266	8-539	8-567	9-608	10-670	10-672	
CRASH	6-278	7-381	8-552	8-557	9-621					
NKRDF\$	#5-58	5-67								
RESRG	#5-58	7-359								
RETURN	6-273	7-294	7-300	7-310	7-321	7-327	7-335	7-342	7-347	7-364
	7-376	7-388	7-396	7-402	7-408	7-415	7-421	7-427	7-439	7-446
	7-465	7-477	7-477	7-483	7-489	7-495	7-501	7-508	8-547	8-575
	9-617	9-632	9-638	9-645	9-651	10-718	11-727	11-732	11-737	11-742
	11-747	11-752	11-757	11-762	11-767	11-772	11-777	12-792	12-805	12-818
	12-831	12-844								
SAVRG	#5-58	7-356								
SLTDF\$	#5-58	5-60								
SOB	6-218	6-223	6-228	6-233	6-238	10-716				
SWSTK\$	6-249	10-670								
TTAB\$	#5-76	5-91	5-92	5-93	5-94	5-95	5-96	5-97	5-98	5-99
	5-100	5-101	5-102	5-103	5-104	5-105	5-106	5-107	5-108	5-109
	5-110	5-111	5-112	5-113	5-114	5-115	5-116	5-117	5-118	5-119
	5-120	5-121	5-122	5-123	5-124	5-125	5-126	5-127	5-128	5-129
	5-130	5-131	5-132	5-133	5-134	5-135	5-136	5-137	5-138	5-139
	5-140	5-141	5-142	5-143						

TTTTTTTTT	MM	MM	PPPPPPP	SSSSSSS	TTTTTTTTT	BBBBBBB	
TTTTTTTTT	MM	MM	PPPPPPP	SSSSSSS	TTTTTTTTT	BBBBBBB	
TT	MMMM	MMMM	PP	SS	TT	BB	BB
TT	MMMM	MMMM	PP	SS	TT	BB	BB
TT	MM	MM	PP	SS	TT	BB	BB
TT	MM	MM	PP	SS	TT	BB	BB
TT	MM	MM	PPPPPPP	SSSSSS	TT	BBBBBBB	
TT	MM	MM	PPPPPPP	SSSSSS	TT	BBBBBBB	
TT	MM	MM	PP	SS	TT	BB	BB
TT	MM	MM	PP	SS	TT	BB	BB
TT	MM	MM	PP	SS	TT	BB	BB
TT	MM	MM	PP	SS	TT	BB	BB
TT	MM	MM	PP	SSSSSSS	TT	BBBBBBB
TT	MM	MM	PP	SSSSSSS	TT	BBBBBBB

LL	SSSSSSS	TTTTTTTTT
LL	SSSSSSS	TTTTTTTTT
LL	SS	TT
LL	SS	TT
LL	SS	TT
LL	SS	TT
LL	SSSSSS	TT
LL	SSSSSS	TT
LL	SS	TT
LL	SS	TT
LL	SS	TT
LL	SS	TT
LLLLLLLLL	SSSSSSS	TT
LLLLLLLLL	SSSSSSS	TT

```

359 .SBTTL .PRI,.POOL,.SCOM,.UMR,.VFY
360 ;
361 ; ".POOL" OR ".SCOM"
362 ;
363 STATES SCOM
364 TRANS !POSEXP,ONLY1,$SCOM
365 TRANS !ERROR,$EXIT
366
367 ; ".PRI"
368 ;
369 STATES PRI
370 TRANS !END,$EXIT,$PRI
371 TRANS %-1%,ONLY1,$PRI1
372 TRANS !ERROR,$EXIT
373
374 ; ".PRTOV"
375 ;
376 STATES PRTOV
377 TRANS !EXP,ONLY1,$PROV
378 TRANS !ERROR,$EXIT
379
380 ; ".UMR"
381 ;
382 STATES UMR
383 .IF DF M$MGE
384 TRANS "ON",ONLY1,$UMRON
385 TRANS "OFF",ONLY1,$UMROF
386 .IFF
387 TRANS "ON",ONLY1
388 TRANS "OFF",ONLY1
389 .ENDC
390 TRANS !ERROR,$EXIT
391
392 ; ".VFY"
393 ;
394 STATES VFY
395 TRANS !POSEXP,ONLY1,$END;
396 TRANS !ERROR,$EXIT
397
398 000002
399 000002
400 000002

```

```

172      BEQ      9$      ;* IF EQ, NOT A MULTIPROCESSOR SYSTEM
173      ASL      R4      ;* CONVERT TO A WORD INDEX
174      5$: SUB      #2,R4 ;* DONE FOR ALL PROCESSORS ?
175      BLT      6$      ;* IF LT, YES .. DIDN'T FIND UNIBUS RUN
176      MOV      URMTB,R2 ;* GET $URMTB ADDRESS
177      ADD      R4,R2    ;* INDEX INTO TABLE
178      BIT      K.URM(R5),(R2) ;* IS CONTROLLER ON THIS CPU ?
179      BEQ      5$      ;* IF EQ, NO
180      BIS      #120000,R0 ;* FORCE APR5 MAPPING
181      BR       7$      ;* AND CONTINUE
182      ;
183      ; FAILED TO FIND UNIBUS RUN
184      ;
185      6$: BIS      #F1.ERR,$FLAG1 ;* DIDN'T FIND UNIBUS RUN (WARNING)
186      BR       40$     ;* AND LEAVE
187      ;
188      ; GOT CPU WITH UNIBUS RUN
189      ;
190      7$: ADD      #K.PRM,R5 ;* POINT R5 AT ICB CHAIN
191      8$: MOV      (R5),R5 ;* GET NEXT ICB IN CHAIN
192      BEQ      101$     ;* IF EQ, NO MORE .. SOMEONE WILL PAY
193      MOV      R5,R2    ;* COPY THE ICB LINK POINTER
194      BIC      #7777,R2 ;* GET CPU NUMBER IN HIGH BITS
195      BIC      R2,R5     ;* CLEAR CPU NUMBER FROM LINK POINTER
196      SWAB     R2      ;* FORM CPU NUMBER*2
197      ASR      R2      ;*
198      ASR      R2      ;* ...
199      ASR      R2      ;* ...
200      MOV      K6TAB,K6TMP ;* GET $K6TAB ADDRESS
201      ADD      R2,K6TMP ;* INDEX INTO TABLE
202      MOV      @K6TMP,@KSAR5 ;* MAP TO CPU PARTITION
203      ASL      R5      ;* FORM VIRTUAL ADDRESS OF ICB
204      MOV      R5,$ICBAD ;* SAVE ICB ADDRESS
205      BIS      #120000,R5 ;* BIAS ICB ADDRESS
206      CMP      R2,R4    ;* IS THIS THE RIGHT CPU ?
207      BNE      8$      ;* IF NE, NO .. KEEP LOOKING
208      9$: MOV      R3,R2 ;* GET NUMBER OF VECTORS
209      10$: MOV     R0,R5 ;* COPY POINTER TO VECTOR
210      BIC      #^C<700>,R5 ;* ISOLATE NONSENSE INTERRUPT NUMBER
211      ASR$     5,R5     ;* AND FORM WORD OFFSET
212      ;
213      CMP      (R0)+,NONS1 ;* IS VECTOR IN USE ?
214      BEQ      11$     ;* IF NOT - BRANCH
215      TST      -(R0)    ;* ELSE - RESET VECTOR POINTER
216      ;
217      ;
218      MOV      NSITAB(R5),NSITMP ;* GET ERROR LOG VECTOR ADDRESS CONTENTS
219      CMP      (R0)+,@NSITMP ;* IS THE VECTOR STILL AVAILABLE?
220      BNE      30$     ;* IF NE, NO
221      TST      (R0)+    ;* SKIP VECTOR PRIORITY
222      SCB      R2,10$   ;* LOOP $NVECT TIMES
223      ;
224      BIS      #F1.VCT,$FLAG1 ;* SHOW THAT WE SETUP THE VECTORS
225      ;
226      .ENDC
227      ;
228      MTPS     #PR7     ;;;* DISABLE INTERRUPTS

```

UNLHC - UNLOAD HOP COST MATRI MACRO V05.03b Saturday 29-Jun-85 00:48 ^{D 12}
Table of contents

5- 45 MACRO DEFINITIONS
6- 56 DEAMHC - DEALLOCATE MINHOP/COST,CKT CTRS.,ROL

C 13

```

LOCAL SYMBOL DEFINITIONS

                                .SBTTL  LOCAL SYMBOL DEFINITIONS
83
84
85
86
87
88
89
90
91
92
93
000002
000004
000006
000010
000012
000014

; SAVED REGISTER OFFSETS FOR SWSTK$
R$R0 = 2
R$R1 = 4
R$R2 = 6
R$R3 = 10
R$R4 = 12
R$R5 = 14

```

D 13

FILEID**UNLSUB

```

UU      UU  NN      NN  LL      SSSSSSSS  UU      UU  BBBB BBBB
UU      UU  NN      NN  LL      SSSSSSSS  UU      UU  BBBB BBBB
UU      UU  NN      NN  LL      SS        UU      UU  BB      BB
UU      UU  NN      NN  LL      SS        UU      UU  BB      BB
UU      UU  NNNN     NN  LL      SS        UU      UU  BB      BB
UU      UU  NNNN     NN  LL      SSSSSS    UU      UU  BBBB BBBB
UU      UU  NN      NN  LL      SSSSSS    UU      UU  BBBB BBBB
UU      UU  NN      NN  LL      SS        UU      UU  BB      BB
UU      UU  NN      NN  LL      SS        UU      UU  BB      BB
UU      UU  NN      NN  LL      SS        UU      UU  BB      BB
UU      UU  NN      NN  LL      SS        UU      UU  BB      BB
UU      UU  NN      NN  LL      SSSSSSSS  UU      UU  BBBB BBBB
UU      UU  NN      NN  LL      SSSSSSSS  UU      UU  BBBB BBBB
UUUUUUUU  NN      NN  LLLLLLLLLL  SSSSSSSS  UUUUUUUUU  BBBB BBBB
UUUUUUUU  NN      NN  LLLLLLLLLL  SSSSSSSS  UUUUUUUUU  BBBB BBBB

```

```

LL      SSSSSSSS  TTTTTTTTTT
LL      SSSSSSSS  TTTTTTTTTT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LL      SSSSSS    TT
LL      SSSSSS    TT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LL      SSSSSSSS  TT
LLLLLLLLLL  SSSSSSSS  TT
LLLLLLLLLL  SSSSSSSS  TT

```

UNLSUB CREATED BY MACRO ON 29-JUN-85 AT 00:48 PAGE 1 C 15
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
DDMAC	= ***** GX	*12-316
DLCAC	= ***** GX	*12-315
FNDADD	000074 RG	#11-234
GETIND	000002 RG	#7-104
GETPDV	000044 RG	#9-176
LF.ACT	= 100000	#5-60
LF.BRO	= 000400	#5-60
LF.BWT	= 000007	#5-60
LF.ENA	= 002000	#5-60
LF.LPB	= 001000	#5-60
LF.MDC	= 000100	#5-60
LF.MFL	= 004000	#5-60
LF.MTP	= 000020	#5-60
LF.PAC	= 000200	#5-60
LF.RDY	= 040000	#5-60 12-303
LF.REA	= 010000	#5-60
LF.SER	= 000040	#5-60
LF.TIM	= 000010	#5-60
LF.UNL	= 020000	#5-60
LF.X2P	= 000000	#5-60
LN.CLO	= 000000	#5-60
LN.DUM	= 000005	#5-60
LN.LOA	= 000004	#5-60
LN.LOO	= 000003	#5-60
LN.OAU	= 000003	#5-60
LN.OFF	= 000001	#5-60
LN.ON	= 000000	#5-60
LN.OOP	= 000004	#5-60
LN.OPE	= 000001	#5-60
LN.REF	= 000002	#5-60
LN.SER	= 000002	#5-60
LN.STA	= 000017	#5-60
LN.SUB	= 000360	#5-60
LN.TRI	= 000036	#5-60
L.COST	000015	#5-60
L.CTL	000012	#5-60 12-311 12-311
L.CVA	177776	#5-60
L.DDM	000002	#5-60 12-308 12-308
L.DDS	000004	#5-60
L.DLC	000003	#5-60 12-305 12-305
L.DLM	000006	#5-60
L.DLS	000010	#5-60
L.FLG	000000	#5-60 12-303
L.KRBA	= 000016	#5-60
L.LEN	= 000022	#5-60
L.MPF	000022	#5-60
L.NMST	000020	#5-60
L.NSTA	000014	#5-60
L.OWNR	000021	#5-60
L.UNT	000013	#5-60
PDVNM	= ***** GX	7-126
PDVTA	= ***** GX	7-108 7-116 9-176

56
57
58 000000
59

: LIBRARY MACROS

.MCALL PCBDF\$
PCBDF\$

; DEFINE PARTITION CONTROL BLOCK OFFSETS

246	000172	112546		XCORE:	MOVB	(R5)+, -(SP)	:	ASSEMBLE BIAS
247	000174	112566	000001		MOVB	(R5)+, 1(SP)	:	...
248	000200	012600			MOV	(SP)+, R0	:	...
249								
250					.IF DF	M\$MGE		
251	000202	122525			CMPB	(R5)+, (R5)+	:	DISCARD DISPLACEMENT
252					.ENDC			
253								
254	000204	112546			MOVB	(R5)+, -(SP)	:	ASSEMBLE COUNT
255	000206	112566	000001		MOVB	(R5)+, 1(SP)	:	...
256	000212	012601			MOV	(SP)+, R1	:	...
257	000214	001402			BEQ	10\$:	IF ZERO, DO NOTHING
258	000216				CALL	\$DEA18	:	DEALLOCATE DOUBLEWORD STORAGE
259	000222			10\$:	RETURN			

```

264          ADD      #4,R5
265          TST      (R3)+
266          .ENDC
267
268 000166          RETURN
269
270          ; CSR
271          ; CSR
272          CSR:      INC      R5
273 000170 005205          TST      (R3)+
274 000172 005723          RETURN
275 000174
276
277          ; COUNTER TIMER
278          ; TIM:
279          TIM:      TST      (R3)+
280 000176 005723          RETURN
281 000200
282
283          ; DEVICE CHARACTERISTICS
284          DVCHA:    TST      FLAGS          ; IS IT MUX EXPANSION ?
285 000202 005767 000154'    BEQ      10$          ; NO
286 000206 001417          L.UNT(R4),INDEX    ; DO UNIT NUMBERS MATCH ?
287 000210 126467 000013 000152'    BNE      10$          ; NO
288 000216 001013          SAVRG      R4          ; SAVE REG
289 000220          MOV      $$SLTA,R4          ; COPY SLT ADDRESS
290 000222 016704 000000G    MOV      R3,L.CVA(R4)    ; COPY FIRST DEVICE CHAR. WORD VIRTUAL ADDR.
291 000226 010364 177776    RESRG      R4          ; RESTORE REG
292 000232
293
294 000234 016723 000000G    MOV      $$DCHA,(R3)+    ; COPY DEVICE CHARACTERISTICS
295 000240 016723 000002G    MOV      $DCHA+2,(R3)+    ; TWO WORDS
296 000244 000401          BR      20$
297 000246 022323          10$:      CMP      (R3)+,(R3)+
298 000250          20$:      RETURN
299
300          ; PROTOCOL EMULATOR CHARACTERISTICS
301          PECHA:    TST      FLAGS          ; IS IT MUX EXPANSION ?
302          BEQ      10$          ; NO
303 000252 005767 000154'    L.UNT(R4),INDEX    ; DO UNIT NUMBERS MATCH ?
304 000256 001407          BNE      10$          ; NO
305 000260 126467 000013 000152'    MOV      $$PCHA,(R3)+    ; COPY PROTOCOL EMULATOR CHARACTERISTICS
306 000266 001003          BR      20$
307 000270 016723 000000G    10$:      TST      (R3)+
308 000274 000401          20$:      RETURN
309 000276 005723
310 000300
311
312          ; END OF FILE
313          EOF:      CRASH
314
315 000302
316
317          ; INTERRUPT ENTRY POINT
318          INT:      CMPB      (R5)+,(R5)+
319
320 000304 122525

```

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES								
ADDRB	000106 R	5-91	#7-220							
ADDRW	000112 R	5-92	#7-226							
APR	001070 R	5-122	#9-265							
BLKB	000116 R	5-93	#7-232							
BLKW	000134 R	5-94	#7-242							
BLOCK	000162 R	#5-160	6-190	10-649						
BYTE	000154 R	5-95	#7-253							
CORE	000160 R	5-96	#7-259							
CSR	000170 R	5-97	#7-273							
CTIM	000176 R	5-98	#7-280							
DPRB	001074 R	5-123	#9-572							
DPRW	001100 R	5-124	#9-578							
DVCHA	000202 R	5-99	#7-285							
EOF	000302 R	5-100	#7-315							
FILE	000530 R	5-101	#8-462							
FLAGS	000154 R	#5-153	*6-181	7-285	7-303	7-365	7-399	*8-461	8-465	*8-483
		*8-511	*9-517	*9-542	12-718	12-731	12-744	12-757	12-770	
FLT	001104 R	#10-595								
FLT16	001104 R	6-186	#10-594							
FL.KMX	= ***** GX	7-404								
INDEX	000152 R	*5-148	*6-180	7-287	7-305	7-367	7-401	8-467	*8-478	8-479
		*8-482	*8-506	8-507	*8-510	9-525	*9-536	9-537	*9-541	12-720
		12-733	12-746	12-759	12-772					
INT	000304 R	5-102	#7-320							
LFILE	000640 R	5-105	#8-495							
LF.ACT	= 100000	#5-62								
LF.BRO	= 000400	#5-62								
LF.BWT	= 000007	#5-62								
LF.ENA	= 002000	#5-62								
LF.LPB	= 001000	#5-62								
LF.MDC	= 000100	#5-62								
LF.MFL	= 004000	#5-62								
LF.MTP	= 000020	#5-62								
LF.PAC	= 000200	#5-62								
LF.RDY	= 040000	#5-62								
LF.REA	= 010000	#5-62								
LF.SER	= 000040	#5-62								
LF.TIM	= 000010	#5-62								
LF.UNL	= 020000	#5-62								
LF.X2P	= 000000	#5-62								
LIBR	000314 R	5-103	#7-333							
LINKS	000324 R	5-104	#7-347							
LN.CLO	= 000000	#5-62								
LN.CUM	= 000005	#5-62								
LN.LOA	= 000004	#5-62								
LN.LOO	= 000003	#5-62								
LN.OAU	= 000003	#5-62								
LN.OFF	= 000001	#5-62								
LN.ON	= 000000	#5-62								
LN.OOP	= 000004	#5-62								
LN.OPE	= 000001	#5-62								
LN.REF	= 000002	#5-62								

```

350 ; COUNTER TIMER
351 ;
352 000274 016724 000222' ;TIM: MOV PRCTIM,(R4)+ ; STORE COUNTER TIMER VALUE
353 000300 RETURN
354 ;
355 ; DEVICE CHARACTERISTICS
356 ;
357 DVCHA: CLR R0 ; ASSUME DLC
358 000302 005000 ;
359 000304 032767 000000G 000162' BIT #FL.DDM,PRFLAG ; IS IT A DDM?
360 000312 001414 BEQ 10$ ; IF EQ, NO
361 000314 032767 000000G 000162' BIT #FL.MUX,PRFLAG ; IS IT A MUX DEVICE?
362 000322 001410 BEQ 10$ ; IF EQ, NO
363 000324 005767 000160' TST FLAGS ; IS IT MUX EXPANSION?
364 000330 100020 BPL 101$ ; IF PL, NO
365 000332 016700 000156' MOV INDEX,R0 ; GET SECONDARY FILE COUNTER
366 000336 006300 ASL R0 ; CONVERT TO A DOUBLEWORD INDEX
367 000340 006300 ASL R0 ;
368 000342 022020 CMP (R0)+,(R0)+ ; SKIP DLC CHARACTERISTICS
369 000344 10$: SAVRG R2 ; SAVE REG
370 000346 016702 000000G MOV $SLTSV,R2 ; COPY SLT ADDRESS
371 000352 010462 177776 MOV R4,L.CVA(R2) ; STORE DEVICE CHAR. POINTER IN SLT
372 000356 RESRG R2 ; RESTORE REG
373 000360 016024 000556' MOV PRDCHA(R0),(R4)+ ; COPY DEVICE CHARACTERISTICS
374 000364 016024 000560' MOV PRDCHA+2(R0),(R4)+ ; TWO WORDS
375 000370 RETURN
376 000372 012724 177777 101$: MOV #-1,(R4)+ ; DDM DVCHA NOT IN MXTAB FILE
377 000376 012724 177777 MOV #-1,(R4)+ ; ...
378 000402 RETURN
379 ;
380 ;
381 ;
382 ;
383 ;
384 ;
385 ;
386 ;
387 ;
388 ;
389 ;
390 ;
391 ;
392 ;
393 ;
394 ;
395 ;
396 ;
397 ;
398 ;
399 000452 EOF: CRASH
400 ;
401 ;
402 ;
403 ;
404 000454 016724 000220' LINKS: MOV PRLINK,(R4)+ ; COPY LOGICAL LINK COUNT
405 000460 RETURN
406 ;

```



```

866
867
868
869
870
871 002170 116724 000512'
872 002174
873
874
875
876
877 002176 016724 000512'
878 002202

```

; DEAD POLLING RATIO BYTE
 DPRB: MOV PRDPR,(R4)+ ; STORE DPR
 RETURN

; DEAD POLLING RATIO WORD
 DPRW: MOV PRDPR,(R4)+ ; STORE DPR
 RETURN

```

53                                     .SBTTL  MACRO DEFINITIONS
54                                     :***
55                                     : MACRO CALLS
56                                     :****
57
58                                     .MCALL  NKRD$,SLTDF$,SAVRG,RESRG
59
60 000000                               SLTDF$                               ; DEFINE SLT OFFSETS AND SYMBOLS
61
62                                     .IF DF  R$$MPL
63                                     .MCALL  KRBD$
64                                     KRBD$                                     ; DEFINE KRB OFFSETS
65                                     .ENDC
66
67 000000                               NKRD$                               ; DEFINE NETWORK KRB OFFSETS
68
69                                     :***
70                                     : LOCAL MACROS
71                                     :***
72
73                                     :
74                                     : DEFINE TRANSFER TABLE ENTRY
75                                     :
76                                     .MACRO  TTAB$  AA,BB
77                                     .WORD   AA
78                                     .ENDM   TTAB$
79
80                                     :***
81                                     : LOCAL DATA
82                                     :***
83
84 000000                               .PSECT  DATA,D
85
86 000000 000000                       TEMP: .WORD  0                               ; TEMP LOCATION
87
88                                     ; TRANSFER TABLE USING FUNCTION CODE AS INDEX
89
90                                     TABLE: TTAB$  ADDR$,NULL                ; EXPANSION ROUTINE IS FIRST ENTRY
91 000002                               TTAB$  ADDR$,NULL                ; CLEANUP ROUTINE IS SECOND ENTRY
92 000004                               TTAB$  BLKB$,POP1
93 000006                               TTAB$  BLKW$,POP1
94 000010                               TTAB$  BYTE$,POP1
95 000012                               TTAB$  CORE$,XCORE
96 000014                               TTAB$  CSR$
97 000016                               TTAB$  CTIM$
98 000020                               TTAB$  DVC$
99 000022                               TTAB$  EOF$,NO
100 000024                               TTAB$  FILE$,POP1
101 000026                               TTAB$  INT$,POP2
102 000030                               TTAB$  LIB$,XLIB$
103 000032                               TTAB$  LINK$,NULL
104 000034                               TTAB$  LFIL$,POP3
105 000036                               TTAB$  LSTHD$,NULL
106 000040                               TTAB$  LTAB$,NULL
107 000042                               TTAB$  MPTAB$,POP3
108 000044                               TTAB$  MXTAB$,POP1
109 000046

```

```
634      ; ACTIVE POLLING RATIO (APR) - BYTE
635      ;
636      ;
637      ; APR:      INC      R3
638      ;          RETURN
639      ;
640      ;
641      ;
642      ; DEAD POLLING RATIO (DPRB) - BYTE
643      ;
644      ; DPRB:     INC      R3
645      ;          RETURN
646      ;
647      ;
648      ;
649      ; DEAD POLLING RATIO (DPRW) - WORD
650      ;
651      ; DPRW:     TST      (R3)+
        ;          RETURN
```

```

TTTTTTTTTT  MM      MM  PPPPPPPP  IIIIII  NN      NN  IIIIII
TTTTTTTTTT  MM      MM  PPPPPPPP  IIIIII  NN      NN  IIIIII
      TT      MMMM  MMMM  PP      PP      II      NN      NN      II
      TT      MMMM  MMMM  PP      PP      II      NN      NN      II
      TT      MM      MM  PP      PP      II      NNNN     NN      II
      TT      MM      MM  PP      PP      II      NNNN     NN      II
      TT      MM      MM  PPPPPPPP  II      NN      NN      NN      II
      TT      MM      MM  PPPPPPPP  II      NN      NN      NN      II
      TT      MM      MM  PP      PP      II      NN      NNNN     II
      TT      MM      MM  PP      PP      II      NN      NNNN     II
      TT      MM      MM  PP      PP      II      NN      NN      II
      TT      MM      MM  PP      PP      II      NN      NN      II
      TT      MM      MM  PP      PP      IIIIII  NN      NN      IIIIII
      TT      MM      MM  PP      PP      IIIIII  NN      NN      IIIIII

```

```

....
....
....
....

```

```

LL      SSSSSSSS  TTTTTTTTTT
LL      SSSSSSSS  TTTTTTTTTT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SSSSSS  TT
LL      SSSSSS  TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LLLLLLLLLL  SSSSSSSS  TT
LLLLLLLLLL  SSSSSSSS  TT

```

5-	56	MACRO DEFINITIONS
6-	62	ENTRY POINT
7-	70	STATE TABLES
8-	106	OPERATOR KEYWORDS STATE TABLES
9-	174	.BIN,.BLKW,.BLKB
10-	212	.BYTE,.CORE
11-	250	.CSR,.END,.FILE
12-	285	.INT,.INT1,.INT2,.INT3,.LIBR,.LFILE
13-	330	.MPTAB,.MXTAB,.MXPTB
14-	359	.PRI,.POOL,.SCOM,.UMR,.VFY
15-	402	.WORD,.X3CHB,.X2CHB,.X3CHW,.X2CHW
16-	438	SUBEXPRESSIONS

```

402 .SBTTL .WORD,.X3CHB,.X2CHB,.X3CHW,.X2CHW
403 ;
404 ; ".WORD"
405 ;
406 STATES WORD
407 TRANS !EXP,WORD1,$WORD
408 TRANS <'>,WORD2,$ZWORD
409 TRANS !ERROR,$EXIT
410 STATES WORD1
411 TRANS !END,$EXIT
412 TRANS <'>
413 STATES WORD2
414 TRANS !EXP,WORD1,$WORD
415 TRANS <'>,WORD2,$ZWORD
416 TRANS !END,$EXIT,$ZWORD
417 ;
418 ; .X3CHB
419 ;
420 STATES X3CHB
421 TRANS !POSEXP,ONLY1,$X3CHB
422 ;
423 ; .X2CHB
424 ;
425 STATES X2CHB
426 TRANS !POSEXP,ONLY1,$X2CHB
427 ;
428 ; .X3CHW
429 ;
430 STATES X3CHW
431 TRANS !POSEXP,ONLY1,$X3CHW
432 ;
433 ; .X2CHW
434 ;
435 STATES X2CHW
436 TRANS !POSEXP,ONLY1,$X2CHW

```

```

229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285

                .IF DF R$$MPL
MOV     FMASK,FMKTMP    ;;;* GET $FMASK ADDRESS
ADD     #2,FMKTMP      ;;;* $FMASK+2 ADDRESS
BIT     #F2,DAS,@FMKTMP ;;;* D-SPACE ENABLED ?
BNE     12$            ;;;* IF YES - BRANCH, WE MUST HAVE AN ICB
CMP     $ADDR,#120000   ;;;* IF NOT - TEST FOR A DSR SPACE LINE TABLE
BLO     20$            ;;;* IF LINE TABLE IS FROM DSR - BRANCH

12$:      ADD     #2,$ICBAD    ;;;* SKIP THE ICB LINK POINTER
15$:      MOV     #PR7,-(R0)   ;;;* SET PRIORITY 7
          MOV     $ICBAD,-(R0) ;;;* COPY ICB ADDRESS
          ADD     -(R1),(R0)   ;;;* ADD IN DISPLACEMENT
          SOB     R3,15$      ;;;* LOOP $NVECT TIMES
          BR      25$         ;;;* CONTINUE

                .IFF

                .IF DF M$$MGE
249 000142 026727 000000G 120000    CMP     $ADDR,#120000    ;;;* PARTITION LINE TABLE?
250 000150 103410                BLO     20$                ;;;* IF LO, NO
251 000152 012740 000000G 15$:      MOV     #PR7,-(R0)   ;;;* SET PRIORITY 7
252 000156 016740 000000G          MOV     $ICBAD,-(R0) ;;;* COPY ICB ADDRESS
253 000162 064110                ADD     -(R1),(R0)   ;;;* ADD IN DISPLACEMENT
254 000164                SOB     R3,15$      ;;;* LOOP $NVECT TIMES
255 0C0170 000407                BR      25$         ;;;* AND CONTINUE

                .ENDC ; DF M$$MGE

                .IFTF
261 000172 012740 000000G 20$:      MOV     #PR7,-(R0)   ;;;* SET PRIORITY 7
262 000176 016740 000000G          MOV     $ADDR,-(R0) ;;;* COPY LINE TABLE ADDRESS
263 000202 064110                ADD     -(R1),(R0)   ;;;* ADD APPROPRIATE DISPLACEMENT
264 000204                SOB     R3,20$      ;;;* LOOP $NVECT TIMES

                .ENDC

268 000210                25$:      MTPS     #0            ;;;* RE-ENABLE INTERRUPTS
269 000214 000403                BR      40$            ;;;*
270 000216 052767 000000G 000000G 30$:      BIS     #F1,VSE,$FLAG1 ;;;* INDICATE ERROR OCCURRED
271 000224                73$:      .IF DF P$$MPL
272                                MOV     (SP)+,@KSARS5 ;;;* RESTORE PREVIOUS MAPPING
273                                .ENDC
274 0C0224                RETURN    ;;;* BACK TO USER MODE

                .IFF

278                                MTPS     #PR7          ;;;* DISABLE INTERRUPTS
279                                MOV     BUFUMP,-(SP)     ;;;* SAVE USER APR2
280                                MOV     KISARO,BUFUMP    ;;;* MAP TO VECTOR SPACE
281                                ADD     #.BASEB,R0       ;;;* ADJUST VECTOR ADDRESS
282                                10$:      CMP     (R0)+,#.EMINT ;;;* IS VECTOR STILL AVAILABLE?
283                                BNE     30$             ;;;* IF NE, NO
284                                TST     (R0)+            ;;;* SKIP VECTOR PRIORITY
285                                SOB     R2,10$          ;;;* LOOP $NVECT TIMES

```

.TITLE UNLHC - UNLOAD HOP COST MATRIX
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

NTL - UNLOAD XPT DATA DASES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0

5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0


```

95                                     .SBTTL $XPSI - UNLOAD PSI DATA BASES
96
97
98                                     +
99                                     $XPSI - UNLOAD PSI DATA BASES FROM EXTENDED POOL
100
101                                     INPUTS:
102                                     NONE
103
104                                     OUTPUTS:
105                                     PSI DATA BASES ARE DE-ALLOCATED
106
107                                     -
108
109                                     .IF NDF I$$AS
110
111 $XPSI:: MOV    @PSIPT,R3          ; GET ADDRESS OF PSI HOME BLOCK
112          BEQ    10$,             ; BR IF NONE - NOTHING TO DO
113          ADD    #H$PVC,R3        ; POINT TO PVC NAME LISTHEAD
114          MOV    #C$LEN,R1        ; GET LENGTH OF PVC NAME BLOCK
115          CALL   DEALST           ; DEALLOCATE PVC NAME BLOCKS
116          ADD    #H$CUG-H$PVC,R3 ; POINT TO CUG NAME LISTHEAD
117          MOV    #G$LEN,R1        ; GET LENGTH OF CUG NAME BLOCK
118          CALL   DEALST           ; DEALLOCATE CUG NAME BLOCK
119          ADD    #H$RDTE-H$CUG,R3 ; POINT TO REMOTE DTE NAME LISTHEAD
120          MOV    #R$LEN,R1        ; GET LENGTH OF REMOTE DTE NAME BLOCK
121          CALL   DEALST           ; DEALLOCATE CUG NAME BLOCK
122          ADD    #H$SVC-H$RDTE,R3 ; POINT TO SVC DESCRIPTOR LISTHEAD
123          MOV    #S$LEN,R1        ; GET LENGTH OF SVC DESCRIPTOR BLOCK
124          CALL   DEALST           ; DEALLOCATE SVC DESCRIPTOR BLOCKS
125          ADD    #H$DST-H$SVC,R3 ; POINT TO X.25 DESTINATION DESCRIPTORS
126          CALL   DEADST           ; DEALLOCATE DESTINATION DESCRIPTORS
127          ADD    #H$D29-H$DST,R3 ; POINT TO X.29 DESTINATION DESCRIPTORS
128          CALL   DEADST           ; DEALLOCATE DESTINATION DESCRIPTORS
129          CALL   XPORT            ; DEALLOCATE PORT TABLE AND CIRCUIT BLOCKS
130          CALL   XLDTE           ; DEALLOCATE LOCAL DTE DESCRIPTOR BLOCKS
131          CALL   XX29            ; DEALLOCATE X.29 DATA BLOCK
132          CALL   XTRA            ; DEALLOCATE TRACE STRUCTURES
133          MOV    @PSIPT,R0        ; GET ADDRESS OF PSI HOME BLOCK
134          MOV    #H$LEN,R1        ; GET LENGTH OF PSI HOME BLOCK
135          CALL   $DEA16          ; DEALLOCATE PSI HOME BLOCK
136          CLR    @PSIPT          ; CLEAR PSI HOME BLOCK ADDRESS
137          RETURN
138          .ENDC                   ; NDF I$$AS
  
```

UNLSUB - NTL UNLOAD SUBROUTINES MACRO V05.03b Saturday 29-Jun-85 00:48 ^{D 14}
Table of contents

5-	52	MACRO DEFINITIONS
6-	68	LOCAL DATA
7-	91	GETIND - GET PDV INDEX
8-	123	FNCIB - FIND CTB ADDRESS
9-	163	GETPDV - CONVERT PDV INDEX TO PDV ADDRESS
10-	181	XKRB - DE-ALLOCATE KRB
11-	221	FNDADD - FIND LINE TABLE ADDRESS
12-	280	SCAN - SCAN SLT FOR ACTIVITY

UNLSUB CREATED BY MACRO ON 29-JUN-85 AT 00:48 PAGE 2 D 15
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
R\$SMPL	= *****	5-62 6-73 8-135 10-198 11-235
R\$S11M	= 000000	10-196
SCAN	000136 RG	#12-295
SF.ACT	= 000200	#5-60
SF.ENA	= 000100	#5-60
SF.LPB	= 000004	#5-60
SF.MFL	= 000040	#5-60
SF.PAC	= 000020	#5-60
SF.REA	= 000010	#5-60
SF.SER	= 000001	#5-60
SF.SVC	= 000002	#5-60
SF.UNL	= 000040	#5-60
SLTMA	= ***** GX	12-298
SLTMM	= ***** GX	12-300
S.COST	000001	#5-60
S.FLG	000000	#5-60
S.LEN	000004	#5-60
S.NMST	000002	#5-60
S.OWNR	000003	#5-60
UNITS	000030 R	#6-88 *12-297 *12-313 12-317
XKRB	000056 RG	#10-195
\$DECEX	= ***** GX	10-203
\$ICBLN	= ***** GX	11-243
\$KRBLN	= ***** GX	10-201
.CBIAS	= ***** GX	9-177

```

61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116

```

```

;+
; $XPARG - RELEASE MCB PROCESS FROM PARTITION
;
; THIS ROUTINE IS CALLED TO FREE THE PROCESS PARTITION. IF IT IS A SYSTEM
; CONTROLLED PARTITION, IT WILL BE UNLINKED FROM THE LIST OF SUB-PCB'S
; AND DE-ALLOCATED.
;
; INPUTS:
;   R0 = PCB ADDRESS
;
; OUTPUTS:
;   C-BIT = SUCCESS/FAILURE
;
;
; .IF DF R$$$11M
$XPARG:
  SWSTK$ 40$
  MOV P.MAIN(R0),R1 ;* ENTER KERNEL MODE
  ;* GET MAIN PARTITION PCB ADDRESS
  .IF NDF R$$$MPL
    BIT #PS.SYS,P.STAT(R1) ;* SYSTEM CONTROLLED MAIN PARTITION?
    BNE 10$ ;* YES
    BICB P.BUSY(R0),P.BUSY+1(R0) ;* UN-BUSY PARTITION
    BICB P.BUSY(R0),P.BUSY+1(R1) ;* UN-BUSY MAIN PARTITION
    CLR P.WAIT+2(R0) ;* ZERO THE ALLOCATION LISTHEAD
  .ENDC ; NDF R$$$MPL
  .IF DF M$$$MGE
  .IF NDF R$$$MPL
    BIT #FE.PLA,@FMASK ;* IS THIS A PLAS SYSTEM ?
    BEQ 40$ ;* NO
    CLR P.PRO(R0) ;* YES .. CLEAR THE PROTECTION WORD
    CR 40$ ;*
  .ENDC ; NDF R$$$MPL
  10$: CMP R0,R1 ;* IS THIS A SUB-PCB?
  BEQ 40$ ;* NO
  20$: CMP P.SUB(R1),R0 ;* DOES THIS SUB-PCB LIN' TO OURS?
  BEQ 30$ ;* YES
  MOV P.SUB(R1),R1 ;* GET NEXT SUB-PCB
  10$ ;* GOT ONE .. GO CHECK IT
  MOV @HEADR,R0 ;* GET USER MODE SP
  MOV (R0),R0 ;*
  INC 6(R0) ;* SET USER MODE C-BIT
  BR 40$ ;*
  30$: MOV P.SUB(R0),P.SUB(R1) ;* UNLINK OUR SUB-PCB
  MOV PLGTH,R1 ;* GET PCB LENGTH
  CALL @DEACB ;* DE-ALLOCATE SUB-PCB
  .ENDC
  40$: RETURN ;* BACK TO USER MODE AND THEN TO CALLER
  .ENDC
  .END
000001

```

```

261
262
263
264 000224
265
266 000224 112546
267 000226 112566 000001
268 000232 012600
269
270 000234 112546
271 000236 112566 000001
272 000242 012602
273 000244 122525
274
275
276
277
278
279 000246

```

```

; LIBRARY POINTER
XLBR:
    .IF DF M$$MGE
    MOVB (R5)+,-(SP) ; ASSEMBLE BIAS
    MOVB (R5)+,1(SP) ; ...
    MOV (SP)+,R0 ; ...
    .IFTF
    MOVB (R5)+,-(SP) ; AND ADDRESS
    MOVB (R5)+,1(SP) ; ...
    MOV (SP)+,R2 ; ...
    CMPB (R5)+,(R5)+ ; DISCARD EXTRA WORD
    .IFF
    MOV R2,R0 ; CALCULATE START OF LIBRARY BLOCK
    BIC #77,R0 ; ...
    .ENDC
    CALLR $XLBR ; RELEASE LIBRARY REFERENCE

```

```

321
322
323 000306 062703 000016      .IF DF M$$MGE
324                                ADD #14.,R3
325                                .IFF
326                                CMP (R3)+,(R3)+
327                                .ENDC
328 000312                      RETURN
329
330      ;
331      ; LIBRARY POINTER
332      ;
333 000314      LIBR:
334                                .IF DF M$$MGE
335                                ADD #6,R5
336                                CMP (R3)+,(R3)+
337                                .IFF
338                                ADD #4,R5
339                                TST (R3)+
340                                .ENDC
341
342 000322                      RETURN
343
344      ;
345      ; LOGICAL LINKS
346      ;
347 000324 005723      LINKS: TST (R3)+
348 000326                      RETURN
349
350      ;
351      ; LISTHEAD
352      ;
353 000330 022323      LSTHD: CMP (R3)+,(R3)+
354 000332                      RETURN
355
356      ;
357      ; LINE TABLE ADDRESS
358      ;
359 000334 005723      LTAB: TST (R3)+
360 000336                      RETURN
361
362      ;
363      ; COMPLEMENT OF PRIORITY
364      ;
365 000340 005767 000154'      PRI: TST FLAGS      ; IS THIS A MUX EXPANSION ?
366 000344 001410                      BEQ 10$      ; IF EQ, NO -- NO UPDATE
367 000346 126467 000013 000152'      CMPB L,UNT(R4),INDEX ; IS THIS THE RIGHT UNIT ?
368 000354 001004                      BNE 10$      ; IF NE, NO
369 000356 012713 000000G      MOV #PR7,(R3)      ; GET COMPLEMENT OF PRIORITY
370 000362 046713 000000G      BIC $$$PR1,(R3)
371 000366 005723      10$: TST (R3)+      ; SKIP PRIORITY
372 000370                      RETURN
373
374      ;
375      ; LOCAL NODE ADDRESS
376      ;
377

```

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES							
LN.SER	= 000002	#5-62							
LN.STA	= 000017	#5-62							
LN.SUB	= 000360	#5-62							
LN.TRI	= 000006	#5-62							
LSTHD	000330 R	5-106	#7-353						
LTAB	000334 R	5-107	#7-359						
L.COST	000015	#5-62							
L.CTL	000012	#5-62							
L.CVA	177776	#5-62	*7-291						
L.CDM	000002	#5-62	6-185						
L.DDS	000004	#5-62	8-469	9-527					
L.DLC	000003	#5-62							
L.DLM	000006	#5-62							
L.DLS	000010	#5-62							
L.FLG	000000	#5-62							
L.KRBA	000016	#5-62							
L.LEN	= 000022	#5-62							
L.MPF	000022	#5-62							
L.NMST	000020	#5-62							
L.NSTA	000014	#5-62							
L.OWNR	000021	#5-62							
L.UNT	000013	#5-62	7-287	7-305	7-367	7-402	8-467	9-525	12-721
		12-747	12-760	12-773					12-734
MPTAB	000636 R	5-108	#8-494						
MXPTB	000722 R	5-126	#9-517						
MXTAB	000524 R	5-109	#8-461						
M\$MGE	= 000000	7-260	7-322	7-334					
NOD	000372 R	5-110	#7-378						
NLPT	= ***** GX	10-596							
PECHA	000252 R	5-127	#7-303						
PR1	000340 R	5-111	#7-365						
PRTOV	000376 R	5-112	#7-385						
PR7	= ***** GX	7-369							
R\$MPL	= *****	5-64							
SCOM	000404 R	5-113	#7-392						
SECSR	000414 R	5-114	#7-399						
SF.ACT	= 000200	#5-62							
SF.ENA	= 000100	#5-62							
SF.LPB	= 000004	#5-62							
SF.MFL	= 000040	#5-62							
SF.PAC	= 000020	#5-62							
SF.REA	= 000010	#5-62							
SF.SER	= 000001	#5-62							
SF.SVC	= 000002	#5-62							
SF.UNL	= 000040	#5-62							
SLNB	000466 R	5-115	#7-418						
SLNW	000472 R	5-116	#7-424						
STNB	000476 R	5-117	#7-430						
STNW	000502 R	5-118	#7-436						
S.COST	000001	#5-62							
S.FLG	000000	#5-62							
S.LEN	000004	#5-62							

```

407
408
409
410 C00462 010400
411 000464 040200
412 000466 056700 000154'
413 000472 005024
414 000474 010024
415 000476
416
417
418
419
420 000500 016724 000164'
421 000504
422
423
424
425
426 000506 005767 000160'
427 000512 100015
428 000514 016700 000156'
429 000520 032767 000000G 000162'
430 000526 001403
431 000530 006200
432 000532 006200
433 000534 006200
434 000536 006300
435 000540 016024 000516'
436 000544
437 000546 012724 177777
438 000552
439
440
441
442
443 000554 017700 000000G
444 000560 016024 000014
445 000564
446
447
448
449
450 000566 112524
451 000570
452
453
454
455
456 000572 005000
457 000574 005767 000160'
458 000600 100003
459 000602 016700 000156'
460 000606 122020
461 000610 116024 000260'
462 000614
463

; LISTHEAD
LSTHD: MOV R4,R0 ; SET REAL ADDRESS
        BIC R2,R0 ; CLEAR APR VALUE
        BIS PRAPR,R0 ; USE THE PROCESS APR
        CLR (R4)+ ; STORE A ZERO WORD
        MOV R0,(R4)+ ; STORE ADDRESS OF PREVIOUS WORD
        RETURN

; LINE TABLE ADDRESS
LTAB: MOV STLIN,(R4)+ ; STORE LINE TABLE STARTING ADDRESS
       RETURN

; SECONDARY CSR
SECSR: TST FLAGS ; IS IT MUX EXPANSION?
        BPL 101$ ; IF PL, NO
        MOV INDEX,R0 ; GET SECONDARY FILE COUNTER
        BIT #FL.KMX,PRFLAG ; IS THIS A COMIOP MUX LINE?
        BEQ 10$ ; IF EQ, NO
        ASR R0 ; DIVIDE COUNTER BY 8
        ASR R0 ; (NUMBER OF LINES PER CONTROLLER)
        ASR R0
        ASL R0 ; CONVERT TO A WORD INDEX
        MOV PRSCSR(R0),(R4)+ ; COPY SECONDARY CSR ADDRESS
        RETURN
10$: MOV PRSCSR(R0),(R4)+ ; SECSR NOT IN MXTAB FILE
     RETURN

; LOCAL NODE ADDRESS
NOD: MOV @DECPT,R0 ; STORE DECNET HOME BLOCK POINTER
      MOV D$LNUM(R0),(R4)+ ; STORE LOCAL NODE ADDRESS
      RETURN

; MAXIMUM PROTOCOL OVERHEAD SIZE
PRTOV: MOV (R5),(R4)+ ; STORE MAX PROTOCOL SIZE
        RETURN

; SYSTEM LINE NUMBER (BYTE)
SLNB: CLR R0 ; ASSUME NORMAL EXPANSION
       TST FLAGS ; IS IT MUX EXPANSION?
       BPL 10$ ; IF PL, NO
       MOV INDEX,R0 ; GET MUX INDEX
       CMPB (R0),(R0)+ ; SKIP DLC LINE NUMBER
       MOV PRSLN(R0),(R4)+ ; STORE SYSTEM LINE NUMBER
       RETURN
10$:

```



```

880          .SBTTL  X3CH
881
882          ;
883          ; DEFAULT BLOCK SIZE
884          ;
885          002204 016724 000224' X3CH01: MOV  PR3DBS,(R4)+ ; STORE DEFAULT BLOCK SIZE
886          002210          RETURN
887          ;
888          ; MAXIMUM BLOCK SIZE
889          ;
890          002212 016724 000226' X3CH02: MOV  PR3MBS,(R4)+ ; STORE MAXIMUM BLOCK SIZE
891          002216          RETURN
892          ;
893          ; DEFAULT WINDOW SIZE
894          ;
895          002220 116724 000230' X3CH03: MOVB PR3DWS,(R4)+ ; STORE DEFAULT WINDOW SIZE
896          002224          RETURN
897          ;
898          ; MAXIMUM WINDOW SIZE
899          ;
900          002226 116724 000232' X3CH04: MOVB PR3MWS,(R4)+ ; STORE MAXIMUM WINDOW SIZE
901          002232          RETURN
902          ;
903          ; CALL TIMER
904          ;
905          002234 116724 000234' X3CH05: MOVB PR3CAL,(R4)+ ; STORE CALL TIMER
906          002240          RETURN
907          ;
908          ; CLEAR TIMER
909          ;
910          002242 116724 000236' X3CH06: MOVB PR3CLR,(R4)+ ; STORE CLEAR TIMER
911          002246          RETURN
912          ;
913          ; RESET TIMER
914          ;
915          002250 116724 000240' X3CH07: MOVB PR3RES,(R4)+ ; STORE RESET TIMER
916          002254          RETURN
917          ;
918          ; RESTART TIMER
919          ;
920          002256 116724 000242' X3CH08: MOVB PR3RST,(R4)+ ; STORE RESTART TIMER
921          002262          RETURN
922          ;
923          ; MAXIMUM CLEAR VALUE
924          ;
925          002264 116724 000244' X3CH09: MOVB PR3MCL,(R4)+ ; STORE MAXIMUM CLEAR VALUE
926          002270          RETURN
927          ;
928          ; MAXIMUM RESETS VALUE
929          ;
930          002272 116724 000246' X3CH10: MOVB PR3MRE,(R4)+ ; STORE MAXIMUM RESETS VALUE
931          002276          RETURN
932          ;
933          ; MAXIMUM RESTARTS VALUE
934          ;
935          002300 116724 000250' X3CH11: MOVB PR3MRS,(R4)+ ; STORE MAXIMUM RESTARTS VALUE
936          002304          RETURN

```

```

MACRO DEFINITIONS

110 000050          TTAB$ NOD,NULL
111 000052          TTAB$ PRI,NULL
112 000054          TTAB$ PRTOV,POP1
113 000056          TTAB$ SCOM,XSCOM
114 000060          TTAB$ SECSR,NULL
115 000062          TTAB$ SLNB,NULL
116 000064          TTAB$ SLNW,NULL
117 000066          TTAB$ STNB,NULL
118 000070          TTAB$ STNW,NULL
119 000072          TTAB$ UNB,NULL
120 000074          TTAB$ UNW,NULL
121 000076          TTAB$ WORD,POP2
122 000100          TTAB$ APR,POP1
123 000102          TTAB$ DPRB,POP1
124 000104          TTAB$ DPRW,POP2
125 000106          TTAB$ TIME,POP2
126 000110          TTAB$ MXPTB,XXPTB
127 000112          TTAB$ PECHA,NULL
128 000114          TTAB$ X3CH01,NULL
129 000116          TTAB$ X3CH02,NULL
130 000120          TTAB$ X3CH03,NULL
131 000122          TTAB$ X3CH04,NULL
132 000124          TTAB$ X3CH05,NULL
133 000126          TTAB$ X3CH06,NULL
134 000130          TTAB$ X3CH07,NULL
135 000132          TTAB$ X3CH08,NULL
136 000134          TTAB$ X3CH09,NULL
137 000136          TTAB$ X3CH10,NULL
138 000140          TTAB$ X3CH11,NULL
139 000142          TTAB$ X2CH01,NULL
140 000144          TTAB$ X2CH02,NULL
141 000146          TTAB$ X2CH03,NULL
142 000150          TTAB$ X2CH04,NULL
143 000152          TTAB$ X2CH05,NULL
144
145          ;
146          ; COPY OF GLOBAL SYMBOLS FROM 'NTLROD'.
147
148 000154          PRSCSR: .BLKW 16.          ; SECONDARY CSR VALUE
149 000214          PRDCHA: .BLKW 2           ; DEVICE CHARACTERISTICS
150 000220          PRSLTA: .BLKW 1           ; SLT ADDRESS
151 000222          PRPRI: .BLKW 1           ; DEVICE PRIORITY
152 000224          PRPCHA: .BLKW 1           ; PROTOCOL EMULATOR CHARACTERISTICS
153 000226          PR2MWS: .BLKW 8.          ; MAXIMUM WINDOW SIZE
154 000246          PR2MBS: .BLKW 8.          ; MAXIMUM BLOCK SIZE
155 000266          PR2MRT: .BLKW 8.          ; MAXIMUM RETRANSMIT COUNT
156 000306          PR2RET: .BLKW 8.          ; RETRANSMIT TIMER
157 000326          PR2HLD: .BLKW 8.          ; HOLDBACK TIMER
158
159          ;
160          ; LOCAL COPY OF $FLAGS
161
162 000346          PRFLAG: .BLKW 1           ; FLAGS WORD
163
164          ;
165          ; SECONDARY FILE INDEX VALUE
166          ;

```

```

653
654
655
656
657
658
659
660
661
662
663
664
665
666 001362
667 001362 005067 000356'
668 001366 017700 000000G
669
670 001372
671
672 001376
673 001402 016211 000000G
674 001406 001467
675 001410 005722
676 001412 005722
677 001414 126204 177776
678 001420 001013
679 001422 022203
680 001424 001442
681 001426 101011
682 001430 011246
683 001432 042716 000003
684 001436 066216 177776
685 001442 022603
686 001444 101032
687 001446 000401
688 001450 005722
689 001452 032722 000001
690 001456 001410
691 001460 005046
692 001462 152216
693 001464 005046
694 001466 152216
695 001470 062616
696 001472 006316
697 001474 006316
698 001476 062602
699 001500 032702 000077
700 001504 001403
701 001506 005712
702 001510 001340
703 001512 000402
704 001514 162702 000100
705 001520 042702 000077
706 001524 011211
707 001526 001330
708 001530 000416
709 001532 005742

;+
;FLT16 - FIND A 16 BIT ALLOCATION LINE TABLE
;
;INPUTS:
;R3=LINE TABLE ADDRESS
;R4=PDV INDEX OF DOM/DLC/LLC PROCESS
;
;OUTPUTS:
;R0,R1,R2,R3=DESTROYED
;UBIAS,UBA=UNLOAD BLOCK ADDRESS
;BLOCK=LOCAL COPY OF UNLOAD BLOCK
;-

FLT16:
FLT: CLR UBA ; INDICATE NO UNLOAD BLOCK FOUND YET
MOV @NTLPT,R0 ; GET NETLDR DATA BLOCK ADDRESS

SWSTK$ 100$ ;* ENTER KERNEL MODE

CALL $MAPX ;* MAP TO THE DATA BLOCK
MOV CXUNL(R2),(R1) ;* AND TO THE FIRST UNLOAD BLOCK
BEQ 100$ ;* IF ZERO, NO UNLOAD BLOCKS
10$: TST (R2)+ ;* SKIP POINTER TO NEXT UNLOAD BLOCK
20$: TST (R2)+ ;* POINT PAST PDV INDEX
CMPB -2(R2),R4 ;* IS THIS THE RIGHT PDV?
BNE 30$ ;* IF NE, NO
CMP (R2)+,R3 ;* IS THIS OUR LINE TABLE?
BEQ 80$ ;* IF EQ, YES
BHI 40$ ;* IF HI, NO
MOV (R2),-(SP) ;* EXTRACT SIZE FIELD
BIC #3,(SP) ;*
ADD -2(R2),(SP) ;* ADD TABLE ENTRY
CMP (SP)+,R3 ;* IS THIS OUR LINE TABLE?
BHI 80$ ;* IF HI, YES
BR 40$ ;* ELSE, SKIP ANY SUB-ALLOCATIONS
30$: TST (R2)+ ;* SKIP THE LINE TABLE ADDRESS
40$: BIT #1,(R2)+ ;* ANY SUB-ALLOCATIONS THIS ENTRY?
BEQ 50$ ;* IF EQ, NO
CLR -(SP) ;* GET NUMBER OF ALLOCATIONS
BISB (R2)+,(SP) ;*
CLR -(SP) ;* AND NUMBER OF LIBRARIES
BISB (R2)+,(SP) ;*
ADD (SP)+,(SP) ;* ADD THEM TOGETHER
ASL (SP) ;* MULTIPLY BY 4
ASL (SP) ;*
ADD (SP)+,R2 ;* GIVING NEXT TABLE ENTRY
50$: BIT #77,R2 ;* IS IT PHYSICAL END OF BLOCK?
BEQ 60$ ;* IF EQ, YES
TST (R2) ;* IS IT LOGICAL END OF BLOCK?
BNE 20$ ;* IF NE, NO - CHECK NEXT ENTRY
BR 70$ ;*
60$: SUB #100,R2 ;* RESET R2 TO START OF BLOCK
70$: BIC #77,R2 ;*
MOV (R2),(R1) ;* MAP TO THE NEXT UNLOAD BLOCK
BNE 10$ ;* IF NE, THERE IS ANOTHER ONE
BR 100$ ;*
80$: TST -(R2) ;* RESET R2 TO FIRST WORD OF ENTRY

```

.TITLE TMPINI - NTL TEMPLATE INITIALIZATION
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

NTL - TEMPLATE PROCESSING INITIALIZATION ROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

.TITLE TMPSTB - NTL TEMPLATE STATE TABLES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

NTL - TEMPLATE SYNTAX TPARS STATE TABLES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

```

438 .SBTTL SUBEXPRESSIONS
439
440 ; EXPRESSION EVALUATION
441 ;
442 STATES$ EXP
443 TRANS$ ',-,NEG,,F2.NEG,$FLAG2
444 TRANS$ $LAMBDA
445 STATES$ POSEXP
446 TRANS$ $NUMBR,$EXIT,$NUM
447 TRANS$ $SYMBOL,$EXIT
448 STATES$ NEG
449 TRANS$ $NUMBR,$EXIT,$NUM
450
451 ; EXTERNAL SYMBOL
452 ;
453 STATES$ SYMBOL
454 TRANS$ $RAD50,$EXIT,$GLOBL
455
456 ; OFFSET MODIFICATION
457 ;
458 STATES$ MODIFY
459 TRANS$ '+,MOD
460 TRANS$ ',-,MOD,,F2.SUB,$FLAG2
461 TRANS$ $LAMBDA,$EXIT,$NOMOD
462 STATES$ MOD
463 TRANS$ $NUMBR,$EXIT,$MOD
464
465 ; FILE NAME
466 ;
467 STATES$ FNAME
468 TRANS$ 'a
469 STATES$ FNAME1
470 TRANS$ $RAD50,$EXIT,$FNAME
471
472 ; MAKE SURE THERE ARE NO OPERANDS
473 ;
474 STATES$ NONE
475 TRANS$ $END,$EXIT
476 TRANS$ $LAMBDA,$EXIT,ERR$3 ; 'NO OPERANDS ALLOWED'
477
478 ; MAKE SURE THERE IS ONLY ONE OPERAND
479 ;
480 STATES$ ONLY1
481 TRANS$ $END,$EXIT
482 TRANS$ $LAMBDA,$EXIT,ERR$4 ; 'ONLY ONE OPERAND ALLOWED'
483
484 ; MAKE SURE THERE ARE ONLY TWO OPERANDS
485 ;
486 STATES$ ONLY2
487 TRANS$ $END,$EXIT
488 TRANS$ $LAMBDA,$EXIT,ERR$5 ; 'ONLY TWO OPERANDS ALLOWED'
489
490
491
492
493
494

```

E 11

```

286      CMP      $ADDR,#60000      ;;; IS LINE TABLE FROM PROCESS SPACE?
287      BHIS     20$                ;;; BR IF NO
288      15$:     MOV     $$PRI,-(R0) ;;; SET DEVICE PRIORITY
289      BIS      #4000,(R0)         ;;; INCLUDE REGISTER SET 1 BIT
290      MOV      $ICBAD,-(R0)       ;;; COPY ICB ADDRESS
291      ADD      -(R1),(R0)         ;;; ADD APPROPRIATE DISPLACEMENT
292      SOB      R3,15$            ;;; LOOP $NVECT TIMES
293      BR       40$                ;;;
294      20$:     MOV     $$PRI,-(R0) ;;; SET DEVICE PRIORITY
295      BIS      #4000,(R0)         ;;; INCLUDE REGISTER SET 1 BIT
296      MOV      $ADDR,-(R0)       ;;; COPY LINE TABLE ADDRESS
297      ADD      -(R1),(R0)         ;;; ADD APPROPRIATE DISPLACEMENT
298      SOB      R3,20$            ;;; LOOP $NVECT TIMES
299      BR       40$                ;;;
300      30$:     BIS      #F1.VSE,$FLAG1 ;;; INDICATE ERROR OCCURRED
301      40$:     MOV      (SP)+,BUFUMP ;;; RESTORE USER APR2
302      MTPS     #0                ;;; RE-ENABLE INTERRUPTS
303
304      .ENDC
305
306      000226      50$:     .IF DF R$$MPL
307      BIT      #F1.ERR,$FLAG1 ; DID WE FIND A UNIBUS RUN ?
308      BEQ      55$                ; IF EQ, YES
309      MSG$     9D                ; ELSE, WARN USER THAT WE DIDN'T FIND UNIBUS RUN
310      BIC      #F1.ERR,$FLAG1 ; NOT A FATAL ERROR !
311
312      55$:     .ENDC
313      000226      032767      000000G 000000G      BIT      #F1.VSE,$FLAG1 ; ANY PROBLEM?
314      000234      001403      BEQ      60$                ; IF EQ, NO
315      000236      MSG$     38                ; VECTOR SETUP ERROR MESSAGE
316      000244      60$:     RETURN
317
318      ; ERROR CONDITIONS
319
320      000246      101$:     CRASH                ; DIDN'T FIND AN ICB FOR THE CPU THAT
321                                ; THIS DEVICE IS ATTACHED TO

```

F 11

45
46
47
48
49
50
51
52
53

000000
000000
000000
000000
000000

.SBTTL MACRO DEFINITIONS

.MCALL ASR\$,ADJDF\$,XPDDB\$,DHBDF\$,CTRDF\$

ADJDF\$; DEFINE ADJACENCY DATA BASE SYMBOLS
DHBDF\$; DEFINE DECNET HOME BLOCK SYMBOLS
XPDDB\$; DEFINE XPT DATA BASE SYMBOLS
CTRDF\$; DEFINE COUNTER SYMBOLS


```

140                                     .SBTTL  DEADST - DEALLOCATE DESTINATION DESCRIPTORS
141
142                                     ;+
143                                     ; DEADST - DEALLOCATE DESTINATION DESCRIPTOR BLOCKS
144                                     ;
145                                     ; INPUTS:
146                                     ; R3 - ADDRESS OF LISTHEAD
147                                     ;
148                                     ; OUTPUTS:
149                                     ; THE DESTINATION DESCRIPTOR BLOCKS ARE DEALLOCATED
150                                     ;
151                                     ; -
152
153                                     .IF NDF 1$$AS
154 DEADST: SWSTK$ 30$                                     ;; ENTER SYSTEM STATE
155 SAVMAP                                     ;; SAVE MAPPING
156 MOV (R3),-(SP)                                     ;; STACK ADDRESS OF FIRST BLOCK
157 CLR (R3)                                     ;; AND ZERO LISTHEAD
158
159 10$: MOV (SP)+,R0                                     ;; GET UNMAPPED ADDRESS OF NEXT BLOCK
160 BEQ 20$                                     ;; IF EQ, END OF LIST
161 MOV R0,-(SP)                                     ;; SET UP FOR $CEACX
162 CALL $CEACX                                     ;; GET ACCESS TO BLOCK
163 MOV (SP)+,R2                                     ;; GET MAPPED ADDRESS OF BLOCK
164 MOV (R2),-(SP)                                     ;; SAVE ADDRESS OF NEXT BLOCK IN LIST
165
166 MOV #D$FLEN,R1                                     ;; GET LENGTH OF FIXED PORTION OF BLOCK
167 CLR -(SP)                                     ;; ZERO WORK CELL ON STACK
168 MOVB D$DATL(R2),(SP)                             ;; GET LENGTH OF CALL MASK (BYTES)
169 ASL (SP)                                     ;; DOUBLE IT FOR CALL DATA
170 ADD (SP),R1                                     ;; UPDATE LENGTH OF BLOCK
171 CLR (SP)                                     ;; ZERO WORK CELL
172 MOVB D$DTEL(R2),(SP)                             ;; GET NUMBER OF DIGITS IN DTE ADDRESS
173 ASR (SP)                                     ;; DIVIDE BY 2 FOR BYTE COUNT
174 ADC (SP)                                     ;;
175 ADD (SP)+,R1                                     ;; UPDATE LENGTH OF BLOCK
176
177 CALL $XDEAC                                     ;; DE-ALLOCATE THE BLOCK
178 BR 10$                                     ;; AND GO AGAIN
179
180 20$: RESMAP                                     ;; RESTORE MAPPING
181 30$: RETURN                                     ;;
182 .ENDC ; NDF 1$$AS

```

E 14

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50

.TITLE UNLSUB - NTL UNLOAD SUBROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

NTL - UNLOAD SUBROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

2.00 06-AUG-81
DECNET-11M/S V3.1
DECNET-11M-PLUS V1.1

3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1

4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0

5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

F 14

UNLSUB CREATED BY MACRO ON 29-JUN-85 AT 00:48 PAGE 3 E 15
MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
CALL	10-202
CALLR	10-203
CRASH	7-114 11-277
NKRDF\$	#5-57 5-59
RETURN	7-121 9-179 10-204 11-273 12-318
SLTDF\$	#5-57 5-60
SOB	12-314
SWSTK\$	10-202

XPARG - RELEASE PARTITION CONTROL MACRO V05.03b Saturday 29-Jun-85 00:49 Page 6-1
Symbol table

AS.DEL= 000010	D\$\$BUG= 177514	L\$\$ASG= 000000	PS.CKP= 040000	P.PRI 000002
AS.EYT= 000004	D\$\$ISK= 000000	L\$\$DRV= 000000	PS.CKR= 020000	P.PRO = ***** GX
AS.RED= 000001	D\$\$L11= 000001	L\$\$P11= 000001	PS.COM= 000200	P.REL 000014
AS.WRT= 000002	D\$\$YNC= 000000	L\$\$11R= 000000	PS.DEL= 000010	P.SIZE 000016
A\$\$CHK= 000000	D\$\$YNM= 000000	M\$\$CRB= 000124	PS.DRV= 000020	P.STAT 000030
A\$\$CPS= 000000	E\$\$XPR= 000000	M\$\$CRX= 000000	PS.FXD= 004000	P.SUB 000010
A\$\$PRI= 000000	FE.PLA= ***** GX	M\$\$FCS= 000000	PS.L10= 001000	P.SWSZ 000022
A\$\$TRP= 000000	FMASK = ***** GX	M\$\$MGE= 000000	PS.NSF= 000400	P.TCB 000026
A.IOC 000003	F\$\$LVL= 000001	M\$\$NET= 000000	PS.OUT= 100000	P.WAIT 000020
A.LGTH= 000014	G\$\$TTP= 000000	M\$\$OVR= 000000	PS.PER= 002000	Q\$\$OPT= 000010
A.MPCT 000011	G\$\$TSS= 000000	N\$\$ACC= 000001	PS.PIC= 000100	R\$\$DER= 000000
A.PCB 000012	G\$\$ITK= 000000	N\$\$BUF= 000001	PS.SYS= 000040	R\$\$K11= 000001
A.PCBL 000000	G\$\$WRD= 000000	N\$\$LDV= 000001	P\$\$P45= 000000	R\$\$SND= 000000
A.PRI 000002	HEADR = ***** GX	N\$\$MCP= 000001	P\$\$WRD= 000000	R\$\$11M= 000000
A.STAT 000010	I\$\$RAR= 000000	N\$\$MML= 000001	P.BLKS 000016	S\$\$WRG= 000000
A.TCB 000004	I\$\$RDN= 000000	N\$\$MOV= 000010	P.BUSY 000024	S\$\$YSZ= 007600
A.TCBL 000006	K\$\$CNT= 177546	N\$\$NCT= 000001	P.IOC 000003	T\$\$KMG= 000000
C\$\$CXP= 000000	K\$\$CSR= 177546	N\$\$PEM= 000001	P.LNK 000000	T\$\$MIN= 000000
C\$\$ORE= 000400	K\$\$LDC= 000000	PLGTH = ***** GX	P.MAIN 000012	V\$\$CTR= 001000
C\$\$RSH= 177564	K\$\$TPS= 000074	PS.APR= 000007	P.NAM 000004	X\$\$DBT= 000000
DEACB = ***** GX	LD\$LP = 000000	PS.CIK= 010000	P.OWN 000026	\$XPARG 000000RG

. ABS. 000032 000 (RW,I,GBL,ABS,OVR)
000132 001 (RW,I,LCL,REL,CON)

Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 9471 Words (37 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:06.05

SY:XPARG.V2,[132,134]XPARG/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]XPARG

```

281
282
283
284 000252 005002
285 000254 152502
286 000256 112546
287 000260 112566
288 000264 012600
289 000266 005702
290 000270 001407
291 000272 005001
292 000274 166701
293 000300 005302
294 000302 001374
295 000304
296 000310
297
298
299
000001
000000G
000001

```

```

; SECONDARY LINKED FILE
XMXPTB: CLR R2
        BISB (R5)+,R2
        MOVB (R5)+,-(SP)
        MOVB (R5)+,1(SP)
        MOV (SP)+,R0
        TST R2
        BEQ 20$
        CLR R1
        SUB $MXLEN,R1
        DEC R2
        BNE 10$
        CALL $DEA16
        RETURN
10$:
20$:

```

```

; GET REPEAT COUNT
; ...
; GET MUX TABLE ADDRESS
; ...
; IS REPEAT COUNT 0 ?
; IF EQ, YES .. DO NOTHING
; CALCULATE LENGTH OF MUX TABLES
; DO IT <FILE COUNT> TIMES
; DEALLOCATE MUX TABLES

```

```

.END

```

```

378 000372 005723      NOD:   TST      (R3)+      ; ADJUST TEMPLATE ADDRESS BY ONE WORD
379 000374                                     ;
380                                     ;
381                                     ;
382                                     ; MAXIMUM PROTOCOL OVERHEAD SIZE
383                                     ;
384                                     ;
385 000376 105723      PRTOV: TSTB      (R3)+      ; ADJUST TEMPLATE ADDRESS BY ONE BYTE
386 000400 105725      TSTB      (R5)+      ; ADJUST BINARY BUFFER POINTER BY ONE BYTE
387 000402      RETURN
388
389                                     ;
390                                     ; POINTER TO ALLOCATED SCOM
391                                     ;
392 000404 062705 000004 SCOM:   ADD      #4,R5
393 000410 005723      TST      (R3)+
394 000412      RETURN
395
396                                     ;
397                                     ; SECONDARY CSR
398                                     ;
399 000414 005767 000154' SECSR:  TST      FLAGS      ; IS IT MUX EXPANSION ?
400 000420 001420      BEQ      20$      ; NO
401 000422 016700 000152'      MOV      INDEX,R0      ; GET SECONDARY FILE COUNTER
402 000426 126400 000013      CMBP     L,UNT(R4),R0      ; DO UNIT NUMBERS MATCH ?
403 000432 001013      BNE      20$      ; NO
404 000434 032767 000000G 000000G      BIT      #FL.KMX,$FLAGS      ; IS THIS A COMIOP MUX LINE ?
405 000442 001403      BEQ      10$      ; NO
406 000444 006200      ASR      R0      ; YES .. DIVIDE COUNTER BY 8
407 000446 006200      ASR      R0      ; (NUMBER OF LINES PER CONTROLLER)
408 000450 006200      ASR      R0
409 000452 006300 10$:   ASL      R0      ; CONVERT TO A WORD INDEX
410 000454 016023 000000G      MOV      $$$(CSR(R0),(R3)+ ; COPY SECONDARY CSR ADDRESS
411 000460 000401      BR       30$
412 000462 005723      20$:   TST      (R3)+
413 000464      30$:   RETURN
414
415                                     ;
416                                     ; SYSTEM LINE NUMBER (BYTE)
417                                     ;
418 000466 005203      SLNB:   INC      R3
419 000470      RETURN
420
421                                     ;
422                                     ; SYSTEM LINE NUMBER (WORD)
423                                     ;
424 000472 005723      SLNW:   TST      (R3)+
425 000474      RETURN
426
427                                     ;
428                                     ; STATION NUMBER (BYTE)
429                                     ;
430 000476 005203      STNB:   INC      R3
431 000500      RETURN
432
433                                     ;
434                                     ; STATION NUMBER (WORD)

```

TMPEX5 CREATED BY MACRO ON 29-JUN-85 AT 00:43 PAGE 3 F 3

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
S.NMST	000002	#5-62
S.OWNR	000003	#5-62
TABLE	000000 R	#5-91 6-198 8-476 8-504 9-534
TIME	001042 R	5-125 #9-553
UBA	000160 R	#5-159 6-187 *10-595 *10-645
UBIAS	000156 R	#5-158 *10-647
UNB	000506 R	5-119 #7-442
UNW	000512 R	5-120 #7-448
WORD	000516 R	5-121 #7-454
X2CH01	001370 R	5-139 #12-718
X2CH02	001424 R	5-140 #12-731
X2CH03	001460 R	5-141 #12-744
X2CH04	001514 R	5-142 #12-757
X2CH05	001550 R	5-143 #12-770
X3CH01	001314 R	5-128 #11-661
X3CH02	001320 R	5-129 #11-666
X3CH03	001324 R	5-130 #11-671
X3CH04	001330 R	5-131 #11-676
X3CH05	001334 R	5-132 #11-681
X3CH06	001340 R	5-133 #11-686
X3CH07	001344 R	5-134 #11-691
X3CH08	001350 R	5-135 #11-696
X3CH09	001354 R	5-136 #11-701
X3CH10	001360 R	5-137 #11-706
X3CH11	001364 R	5-138 #11-711
ZTIME	= ***** GX	9-553
\$ADDR	= ***** GX	6-183
\$BIN	= ***** GX	6-182
\$FLAGS	= ***** GX	7-404
\$MAPX	= ***** GX	10-599
\$PTR	= ***** GX	6-192
\$SLTA	= ***** GX	6-184 6-191 7-290
\$TMEX5	000000 RG	#6-179
\$X2HLD	= ***** GX	12-776
\$X2MBS	= ***** GX	12-737
\$X2MRT	= ***** GX	12-750
\$X2MWS	= ***** GX	12-724
\$X2RET	= ***** GX	12-763
\$SDCHA	= ***** GX	7-294 7-295
\$SPCHA	= ***** GX	7-307
\$SPRI	= ***** GX	7-370
\$SSCSR	= ***** GX	7-410
.CXUNL	= ***** GX	10-600
.EGF	= ***** GX	8-473 8-501 9-531

```

464
465      ; SYSTEM LINE NUMBER (WORD)
466
467      SLNW: CLR      R0      ; ASSUME NORMAL EXPANSION
468            TST      FLAGS   ; IS IT MUX EXPANSION?
469            BPL      10$     ; IF PL, NO
470            MOV      INDEX,R0 ; GET MUX INDEX
471            CMPB     (R0)+,(R0)+ ; SKIP DLC LINE NUMBER
472            CLR      R3      ; GET LINE NUMBER AS A WORD
473            BISB     PRSLN(R0),R3 ; STORE it
474            MOV      R3,(R4)+
475            RETURN
476
477      ; STATION NUMBER (BYTE)
478
479
480      STNB: TSTB     FLAGS   ; IS IT MTP EXPANSION?
481            BMI      10$     ; IF M1, YES
482            CMP      #1,PRMTP ; ARE WE DEALING WITH ONLY 1 STATION ?
483            BNE      101$    ; IF NE, NO
484            MOV      INDEX,R0 ; GET CURRENT STATION INDEX
485            MOVB     PRSNUM(R0),(R4)+ ; STORE STATION NUMBER
486            RETURN
487            MOVB     #-1,(R4)+ ; DLC STNB NOT IN MPTAB FILE
488            RETURN
489
490      ; STATION NUMBER (WORD)
491
492
493      STNW: TSTB     FLAGS   ; IS IT MTP EXPANSION?
494            BMI      10$     ; IF M1, YES
495            CMP      #1,PRMTP ; ARE WE DEALING WITH ONLY 1 STATION ?
496            BNE      101$    ; IF NE, NO
497            MOV      INDEX,R0 ; GET CURRENT STATION INDEX
498            CLR      R3      ; GET STATION NUMBER
499            BISB     PRSNUM(R0),R3 ; STORE it
500            MOV      R3,(R4)+
501            RETURN
502            MOV      #-1,(R4)+ ; DLC STNW NOT IN MPTAB FILE
503            RETURN
504
505      ; UNIT NUMBER (BYTE)
506
507
508      UNB:  TST      FLAGS   ; IS IT EITHER MUX OR MTP EXPANSION?
509            BEQ      101$    ; IF EQ, NO
510            MOVB     INDEX,(R4)+ ; STORE CURRENT MUX INDEX
511            RETURN
512            MOVB     #-1,(R4)+ ; UNB NOT IN EITHER MXTAB OR MPTAB FILE
513            RETURN
514
515      ; UNIT NUMBER (WORD)
516
517
518      UNW:  TST      FLAGS   ; IS IT EITHER MUX OR MTP EXPANSION?
519            BEQ      101$    ; IF EQ, NO
520            MOV      INDEX,(R4)+ ; STORE CURRENT MUX INDEX

```



```

938 .SBTTL X2CH
939
940 : MAXIMUM WINDOW SIZE
941
942 X2CH01: CLR R0 ; ASSUME NOT A MUX EXPANSION
943 002306 005000 BIT #FL.MUX,$FLAGS ; IS IT A MUX DEVICE?
944 002310 032767 BEQ 10$ ; IF EQ, NO
945 002316 001403 MOV INDEX,R0 ; GET SECONDARY FILE COUNTER
946 002320 016700 ASL R0 ; CONVERT TO A WORD INDEX
947 002324 006300 10$: MOVB PR2MWS(R0),(R4)+ ; STORE MAXIMUM WINDOW SIZE
948 002326 116024 001226' RETURN
949
950 : MAXIMUM BLOCK SIZE
951
952 X2CH02: CLR R0 ; ASSUME NOT A MUX EXPANSION
953 002334 005000 BIT #FL.MUX,$FLAGS ; IS IT A MUX DEVICE?
954 002336 032767 BEQ 10$ ; IF EQ, NO
955 002344 001403 MOV INDEX,R0 ; GET SECONDARY FILE COUNTER
956 002346 016700 ASL R0 ; CONVERT TO A WORD INDEX
957 002352 006300 10$: MOVB PR2MBS(R0),(R4)+ ; STORE MAXIMUM BLOCK SIZE
958 002354 016024 001246' RETURN
959
960 : MAXIMUM RETRANSMIT COUNT
961
962 X2CH03: CLR R0 ; ASSUME NOT A MUX EXPANSION
963 002362 005000 BIT #FL.MUX,$FLAGS ; IS IT A MUX DEVICE?
964 002364 032767 BEQ 10$ ; IF EQ, NO
965 002372 001403 MOV INDEX,R0 ; GET SECONDARY FILE COUNTER
966 002374 016700 ASL R0 ; CONVERT TO A WORD INDEX
967 002400 006300 10$: MOVB PR2MRT(R0),(R4)+ ; STORE MAXIMUM RETRANSMIT COUNT
968 002402 116024 001266' RETURN
969
970 : RETRANSMIT TIMER
971
972 X2CH04: CLR R0 ; ASSUME NOT A MUX EXPANSION
973 002410 005000 BIT #FL.MUX,$FLAGS ; IS IT A MUX DEVICE?
974 002412 032767 BEQ 10$ ; IF EQ, NO
975 002420 001403 MOV INDEX,R0 ; GET SECONDARY FILE COUNTER
976 002422 016700 ASL R0 ; CONVERT TO A WORD INDEX
977 002426 006300 10$: MOVB PR2RET(R0),(R4)+ ; STORE RETRANSMIT TIMER
978 002430 016024 001306' RETURN
979
980 : HOLDBACK TIMER
981
982 X2CH05: CLR R0 ; ASSUME NOT A MUX EXPANSION
983 002436 005000 BIT #FL.MUX,$FLAGS ; IS IT A MUX DEVICE?
984 002440 032767 BEQ 10$ ; IF EQ, NO
985 002446 001403 MOV INDEX,R0 ; GET SECONDARY FILE COUNTER
986 002450 016700 ASL R0 ; CONVERT TO A WORD INDEX
987 002454 006300 10$: MOVB PR2HLD(R0),(R4)+ ; STORE HOLDBACK TIMER
988 002462 016024 001326' RETURN
989
990 000001 .END
991

```

```

167 000350          INDEX: .BLKW 1
168
169          ;
170          ; MUX/MTP FLAGS
171
172 000352          FLAGS: .BLKW 1          ; ZERO = NORMAL, NON-ZERO = MUX
173
174          ;
175          ; DATA NEEDED FOR SEARCH OF UNLOAD BLOCKS
176
177 000354          UBIAS: .BLKW 1          ; BIAS/ADDRESS OF LIBRARY DESCRIPTOR
178 000356          UBA: .BLKW 1          ; OFFSET TO LIBRARY DESCRIPTOR
179 000360          BLOCK: .BLKW 32.        ; LOCAL COPY OF UNLOAD BLOCK (FROM POOL..)
180
181 000000          .PSECT
    
```

MACRO DEFINITIONS

710	001534	010267	000356'	MOV	R2,UBA	;* SAVE UNLOAD ENTRY ADDRESS
711	001540	011167	000354'	MOV	(R1),UBIAS	;* AND BIAS
712	001544	042702	000077'	BIC	#77,R2	;* RESET R2 TO START OF BLOCK
713	001550	012700	000360'	MOV	#BLOCK,R0	;* MAKE A LOCAL COPY OF THE UNLOAD BLOCK
714	001554	012703	000040'	MOV	#32,R3	;* ...
715	001560	012220		MOV	(R2)+(R0)+	;* ...
716	001562			SOB	R3,90\$;* ...
717						
718	001566			90\$:	RETURN	;* BACK TO USER MODE AND TO CALLER
719				100\$:		

```

54      ;***
55      ; LIBRARY MACROS
56      ;***
57      .MCALL  MSG$,NTL$,NKRDF$,MSG$,SLTDF$
58
59      000000      SLTDF$      ; DEFINE SLT OFFSETS
60
61      .IF DF  R$$MPL
62      .MCALL  KRBDF$,STAF$,RMAF$,DIR$
63      KRBDF$      ; DEFINE KRB OFFSETS
64      SETAF:  STAF$  0      ; SET AFFINITY
65      .ENDC
66
67      000000      NKRDF$      ; DEFINE NETWORK KRB OFFSETS
68
69      ;***
70      ; LOCAL DATA
71      ;***
72
73      000000      .PSECT  DATA,D
74
75      .NLIST  BEX
76
77      ;
78      ; TEMPLATE EXTENSION
79      ;
80      000000      104      101      124  TMPEXT: .ASCIZ  "DAT"
81
82      ;
83      ; ERROR MESSAGES
84      ;
85      .ENABL  LC
86
87      000004      NTLR$  .8,3,$TWE,$PRV.N,$FNOUT,<Open Failure (-***.)>
88      000042      NTLR$  .WZ,,,$TWARN,RTSPC,,<MUX update only>
89
90      .DSABL  LC
91
92      ;
93      ; ERROR MESSAGE FORMAT STRING
94      ;
95      .ENABL  LC
96      .NLIST  BEX
97      000072      000      040      124  FMT0: .ASCIZ  ""
98      000073      052      FMT3: .ASCIZ  "* Template File -- "
99      .EVEN
100     .DSABL  LC
101     .LIST  BEX
102
103     000000      .PSECT

```

56
57
58
59
60

.SBTTL MACRO DEFINITIONS
;***
; LIBRARY MACROS
;***
.MCALL ISTAT\$,STAT\$,TRANS

```

495
496
497
498
499 000002
500 000002
501 000002
502
503
504
505
506
507
508
509
510
511
512
513
514
515 000002
516 000002
517 000002
518
519
520
521
522 000002
523
524
525 000001

; OPERAND ERROR
;
; STATES$ ERROR
; TRANS$ 'END,$EXIT,ERR$6 ; 'OPERAND MISSING'
; TRANS$ $LAMBDA,$EXIT,ERR$7 ; 'ILLEGAL OPERAND'
;
; "EAT" A POSITIVE EXPRESSION
;
; .IF NDF M$$MGE
; STATES$ EAT
; TRANS$ $NUMBR,$EXIT
; TRANS$ $RAD50,$EXIT
; .ENDC
;
; CHECK FOR END OF SOURCE LINE
;
; STATES$ END
; TRANS$ <'>,$EXIT
; TRANS$ $EOS,$EXIT
;
; LAST STATE
;
; STATES$
;
; .END

```

```

323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339 000250
340 000250 005767 000000G
341 000254 100002
342 000256 000167 000300
343 000262 016700 000000G 2$:
344 000266 032767 000000G 000000G
345 000274 001130
346
347
348 000276 032767 000000G 000000G
349 000304 001417
350
351 000306 022767 120000 000000G
352 000314 101013
353 000316 016701 000000G
354 000322 032767 000000G 000000G
355 000330 001402
356 000332 016701 000000G
357 000336 016160 000000G 000006 3$:
358
359
360 000344 016701 000000G 5$:
361 000350 001504
362 000352 066701 000000G
363 000356 032767 000000G 000000G
364 000364 001011
365 000366 032767 000000G 000000G
366 000374 001066
367 000376 016700 000000G
368 000402 010160 000000G
369 000406 000465
370 000410 032767 000000G 000000G 10$:
371 000416 001447
372 000420 005767 000000G
373 000424 001402
374 000426 016701 000000G
375 000432
376 000432 016746 000000G
377 000436 062716 000000G
378 000442 017616 000000G
379

```

+ \$TMLTA - SETUP DATA BASE ADDRESS IN SYSTEM LINE TABLE
 INPUTS:
 \$ADDR = DATA BASE ADDRESS
 \$MXADD = MUX TABLES ADDRESS
 \$OFF = OFFSET VALUE
 \$SLTA = SYSTEM LINE TABLE ADDRESS
 FL.ERR SET IF ANY ERRORS OCCURRED
 OUTPUTS:
 RO,R1=DESTROYED
 L.DDS AND/OR L.DLS SET TO DATA BASE ADDRESS
 L.DLM SET TO BIAS OF DLC DATA BASE IF DATA BASE IS MAPPED
 LF.RDY SET IF APPROPRIATE

```

$TMLTA::
    TST $FLAG1 ; HAVE THERE BEEN ANY ERRORS ?
    BPL 2$ ; BR IF NO
    JMP 80$ ; ELSE DO NOTHING
    MOV $SLTA,R0 ; GET SYSTEM LINE TABLE ADDRESS
    BIT #FL.CHA,$FLAGS ; MUX CHARACTERISTICS UPDATE ONLY ?
    BNE 70$ ; YES .. GO SET THE READY BIT

    IF DF M$MGE
    BIT #FL.DLC,$FLAGS ; IS THIS PROCESS A DLC ?
    BEQ 5$ ; BR IF NO, NORMAL DATA BASE LOAD

    CMP #120000,$ADDR ; IS THE LINE TABLE MAPPED ?
    BHI 5$ ; IF HI, NO - NORMAL LOAD
    MOV $DLCPU,R1 ; ASSUME IT'S NOT A DDM ALSO
    BIT #FL.DDM,$FLAGS ; IS THIS A DDM ALSO ?
    BEQ 3$ ; IF EQ, NO
    MOV $PDVA,R1 ; GET THE PDV ADDRESS FOR THIS PROCESS
    MOV Z.DSP(R1),L.DLM(R0) ; SET THE DATA BASE BIAS
    .ENDC

    MOV $ADDR,R1 ; GET DATA BASE ADDRESS
    BEQ 80$ ; IF EQ, DO NOTHING
    ADD $OFF,R1 ; ADD OFFSET VALUE
    BIT #FL.DDM,$FLAGS ; ARE WE LOADING A DDM?
    BNE 10$ ; IF NE, YES
    BIT #FL.DLC,$FLAGS ; ARE WE LOADING A DLC?
    BNE 60$ ; IF NE, YES
    MOV $PDVA,R0 ; GET PDV ADDRESS
    MOV R1,Z.DAT(R0) ; SET LLC DATA BASE ADDRESS
    BR 80$

    BIT #FL.MUX,$FLAGS ; IS THIS A MUX LINE?
    BEQ 50$ ; IF EQ, NO
    TST $MXADD ; MUX TABLES SEPARATE ?
    BEQ 15$ ; IF EQ, NO
    MOV $MXADD,R1 ; ELSE, POINT TO MUX TABLES

    MOV SLINM,-(SP) ; GET NUMBER OF SLT ENTRIES ON STACK
    ADD #AB,(SP) ; ...
    MOV a(SP),(SP) ; VECTORED EXEC ACCESS TO $SLINM+AB

```

```

55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71 000000 005763 000072 $DEAHC::TST N$ADJ1(R3) ; GET ADJACENCY DATA BASE BIAS
72 000004 001441 BEQ 5$ ; IF NONE - BRANCH
73 000006 017702 000000G MOV @DECP1,R2 ; GET DECNET HOME BLOCK ADDRESS
74 000012 016201 000030 MOV D$NLN(R2),R1 ; COPY NUMBER OF CIRCUITS
75 000016 066201 000054 ADD D$NBRA(R2),R1 ; INCLUDE NUMBER OF ROUTER ADJACENCIES
76 000022 066201 000056 ADD D$NBEA(R2),R1 ; INCLUDE NUMBER OF ENDNODE ADJACENCIES
77 000026 012700 000004 MOV #A$LEN,R0 ; STORE SIZE
78 000032 CALL $MUL ; CALCULATE TOTAL SIZE
79 000036 062701 000077 ADD #77,R1 ; ROUND UP
80 000042 042701 000077 BIC #77,R1 ; ...
81 000046 ASR$ 6,R1 ; CONVERT TO BLOCKS
82 000062 016300 000072 MOV N$ADJ1(R3),R0 ; COPY ADJACENCY BIAS
83 000066 C10146 MOV R1,-(SP) ; SAVE NUMBER OF BLOCKS
84 000070 CALL $DEA18 ; DEALLOCATE BLOCK
85
86 000074 012601 MOV (SP)+,R1 ; GET NUMBER OF BLOCKS
87 000076 016300 000074 MOV N$ADJ2(R3),R0 ; GET SECOND ADJACENCY DATA BASE BIAS
88 000102 001402 BEQ 5$ ; IF NONE - BRANCH
89 000104 CALL $DEA18 ; DEALLOCATE BLOCK
90
91 000110 016301 000036 5$: MOV N$MHC1(R3),R1 ; GET NO. OF BYTES IN NODE MINHOP/COST
92 000114 066301 000044 ADD N$MHC2(R3),R1 ; AND AREA MINHOP/COST MATRIX
93 000120 006301 ASL R1 ; CONVERT TO BYTES
94 000122 062701 000077 ADD #77,R1 ; ROUND TO NEXT BLOCK BOUNDARY
95 000126 042701 000077 BIC #77,R1 ; ...
96 000132 ASR$ 6,R1 ; CONVERT TO BLOCK COUNT
97 000146 016300 000040 MOV N$MHC1+2(R3),R0 ; GET APR BIAS
98 000152 001402 BEQ 10$ ; NO TABLE ALLOCATED
99 000154 CALL $DEA18 ; DEALLOCATE NODE COUNTERS-MINHOP/COST
100
101 000160 012700 000046 10$: MOV #T$LEN,R0 ; GET COUNTER SIZE
102 000164 016301 000100 MOV N$TLC(R3),R1 ; GET NUMBER OF LINE COUNTERS
103 000170 CALL $MUL ; CALCULATE BLOCK SIZE
104 000174 062701 000077 ADD #77,R1 ; ROUND UP TO NEXT BOUNDARY
105 000200 042701 000077 BIC #77,R1 ; ...
106 000204 ASR$ 6,R1 ; GET BLOCK COUNT
107 000220 016300 000102 MOV N$TLC+2(R3),R0 ; GET APR BIAS
108 000224 001402 BEQ 15$ ; IF NONE BRANCH
109 000226 CALL $DEA18 ; DEALLOCATE BLOCK
110
111 000232 016301 000022 15$: MOV N$ROA1(R3),R1 ; GET NUMBER OF ROL ENTRIES

```


F 13

```

184                                     .SBTTL XLDTE - DEALLOCATE LOCAL DTE BLOCK
185
186                                     :+
187                                     :XLDTE - DEALLOCATE LOCAL DTE DESCRIPTOR BLOCK
188                                     :
189                                     :INPUTS:
190                                     :   NONE
191                                     :
192                                     :OUTPUTS:
193                                     :   LOCAL DTE DESCRIPTOR BLOCKS, CHANNEL HASH TABLES, AND
194                                     :   OUTGOING CHANNEL LISTS ARE DEALLOCATED
195                                     :
196                                     :-
197
198                                     .IF NDF I$$$S
199
200 XLDTE:                                SWSTK$ 50$                :: ENTER SYSTEM STATE
201                                SAVMAP                :: SAVE PREVIOUS MAPPING
202
203
204 000260 017703 000000G 10$: MOV    APSIPT,R3                :: GET PSI HOME BLOCK ADDRESS
205 000264 016300 000002  MOV    H$LDTE(R3),R0            :: GET NEXT BLOCK IN LIST
206 000270 001457                BEQ    40$                :: BR IF END OF LIST
207 000272 010046                MOV    R0,-(SP)            :: GET UNMAPPED ADDRESS OF BLOCK
208 000274                CALL    $CEACX                    :: GET ACCESS TO BLOCK
209 000300 012604                MOV    (SP)+,R4            :: RETRIEVE MAPPED ADDRESS OF BLOCK
210 000302 011463 000002  MOV    (R4),H$LDTE(R3)        :: UNLINK FROM LIST
211
212                                SAVRG <R0>                :: SAVE R0 (UNMAPPED ADDRESS OF BLOCK)
213 000310                                SAVRG <#L$LEN>        :: STACK LENGTH TO DEALLOCATE
214
215 000314 016400 000034 15$: MOV    L$CHLS(R4),R0            :: GET OUTGOING CHANNELS LIST ADDRESS
216 000320 001407                BEQ    20$                :: BR IF NONE
217 000322 011064 000034  MOV    (R0),L$CHLS(R4)        :: SAVE LINK TO NEXT
218 000326 012701 000006  MOV    #K$LEN,R1                :: GET LENGTH OF OUTGOING CHANNELS LIST
219 000332                CALL    $DECEX                    :: DEALLOCATE THE BLOCK
220 000336 000766                BR    15$                :: GO AGAIN
221
222 000340 016400 000030 20$: MOV    L$CHTB(R4),R0            :: GET HASH TABLE ADDRESS
223 000344 001424                BEQ    30$                :: BR IF NONE
224 000346 016401 000032  MOV    L$(TEN(R4),R1            :: GET NUMBER OF ENTRIES IN HASH TABLE
225 000352                ASL$ 2,R1                :: CALCULATE SIZE (4 BYTES PER ENTRY)
226
227 000356 010402                MOV    R4,R2                :: COPY MAPPED LOCAL DTE ADDRESS
228 000360 062702 000122  ADD    #L$LEN,R2                :: COMPUTE ADDRESS OF END OF DTE
229 000364 020027 120000  CMP    R0,#120000                :: IS THIS AN XPOOL ADDRESS?
230 000370 103404                BLO    23$                :: BR IF NO
231 000372 042700 160000  BIC    #160000,R0            :: ELSE SET APR5 BIAS
232 000376 052700 120000  BIS    #120000,R0            ::
233 000402 020002 23$: CMP    R0,R2                :: IS HASH TABLE CONTIGUOUS WITH DTE?
234 000404 001403                BEQ    25$                :: YES, DEALLOCATE TOGETHER
235 000406                CALL    $DECEX                    :: ELSE DEALLOCATE HASH TABLE TO POOL
236 000412 000401                BR    30$                :: MERGE
237
238 000414 060176 25$: ADD    R1,(SP)                :: INCLUDE LENGTH OF HASH TABLE IN DEALLOCATION
239 000416 30$: RESRG <R1>                :: GET LENGTH TO DEALLOCATE
240 000420                RESRG <R0>                :: RETRIEVE ADDRESS OF LOCAL DTE DESCRIPTOR

```

G 13

```

52
53
54
55
56
57
58
59 000000
60 000000
61
62
63
64
65
66

.SBTTL MACRO DEFINITIONS

;
; LIBRARY MACROS
;
.MCALL NKRDF$,SLTDF$
NKRDF$ ; DEFINE NETWORK KRB OFFSETS
SLTDF$ ; DEFINE SLT OFFSETS

; IF DF R$$$MPL
.MCALL KR3DF$
KR3DF$ ; DEFINE KRB OFFSETS
.ENDC

```


XPAR CREATED BY MACRO ON 29-JUN-85 AT 00:49 PAGE 1 F 16
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
DEACB	= ***** GX	6-110
FE.PLA	= ***** GX	6-91
FMASK	= ***** GX	6-91
HEADR	= ***** GX	6-104
M\$MGE	= 000000	6-89
PLGTH	= ***** GX	6-109
PS.SYS	= 000040	6-81
P.BUSY	000024	6-83 *6-83 6-84 *6-84
P.MAIN	000012	6-77
P.PRO	= ***** GX	*6-93
P.STAT	000030	6-81
P.SUB	000010	6-100 6-102 6-108 *6-108
P.WAIT	000020	*6-85
R\$MPL	= *****	6-79 6-90
R\$B11M	= 000000	6-74
\$XPAR	000000 RG	#6-75

```

Symbol table
A$$CHK= 000000      G$$ITK= 000000      M$$MGE= 000000      P$$WRD= 000000      XLIBR  000224R
A$$CPS= 000000      G$$WRD= 000000      M$$NET= 000000      P.REL = ***** GX      XMXPB  000252R
A$$PRI= 000000      I$$RAR= 000000      M$$DVR= 000000      P.WAIT= ***** GX      XSCDM  000052R
A$$TRP= 000000      I$$RDN= 000000      NULL  000050R      Q$$OPT= 000010      X$$DBT= 000000
C$$CKP= 000000      K$$CNT= 177546      N$$ACC= 000001      R$$DER= 000000      $BIN  = ***** GX
C$$ORE= 000400      K$$CSR= 177546      N$$BUF= 000001      R$$K11= 000001      $DEAC2= ***** GX
C$$RSH= 177564      K$$LDC= 000000      N$$LDV= 000001      R$$SND= 00000C      $DEA16= ***** GX
D$$BUG= 177514      K$$TPS= 000074      N$$MCP= 000001      R$$11M= 00000C      $DEA18= ***** GX
D$$ISK= 000000      LD$LP = 000000      N$$MML= 000001      S$$WRG= 000000      $MAPX = ***** GX
D$$L11= 000001      L$$ASG= 000000      N$$MOV= 000010      S$$YSZ= 007600      $MXLEN= ***** GX
D$$SYN= 000000      L$$DRV= 000000      N$$NCT= 000001      TABLE 000000R      $PCB  = ***** GX
D$$SYN= 000000      L$$P11= 000001      N$$PEM= 000001      T$$KMG= 00000C      $PRAVL= ***** GX
E$$XPR= 000000      L$$11R= 000000      POP1  000044R      T$$MIN= 000000      $PTR  = ***** GX
F$$LVL= 000001      M$$CRB= 000124      POP2  000040R      V$$CTR= 001000      $TMEX3 000000RG
G$$TTP= 000000      M$$CRX= 000000      POP3  000032R      XCORE  000172R
G$$TSS= 000C00      M$$FCS= 000000      P$$P45= 000000

```

```

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
DATA 000312 C01 (RW,I,LCL,REL,CON)
DATA 000152 C02 (RW,D,LCL,REL,CDN)
Errors detected: 0

```

*** Assembler statistics

```

Work file reads: 0
Work file writes: 0
Size of work file: 8968 Words ( 36 Pages)
Size of core pool: 14440 Words ( 55 Pages)
Operating system: RSX-11M/PLUS

```

Elapsed time: 00:00:08.43
SY:TMPEX3.V2,[132,134]TMPEX3/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]TMPEX3

435			:		
436	000502	005723	STNW:	TST	(R3)+
437	000504			RETURN	
438			:		
439			:		
440			:	UNIT NUMBER (BYTE)	
441			:		
442	000506	005203	UNB:	INC	R3
443	000510			RETURN	
444			:		
445			:		
446			:	UNIT NUMBER (WORD)	
447			:		
448	000512	005723	UNW:	TST	(R3)+
449	000514			RETURN	
450			:		
451			:		
452			:	SINGLE WORD	
453			:		
454	000516	122525	WORD:	CMPB	(R5)+,(R5)+
455	000520	005723		TST	(R3)+
456	000522			RETURN	

TMPEX5 CREATED BY MACRO ON 29-JUN-85 AT 00:43 PAGE 4 6 3

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

CALL	6-186	6-198	8-476	8-504	9-534	10-597	10-599			
CRASH	6-205	7-315	8-489	8-494	9-547					
NKRDF\$	#5-60	5-69								
RESRG	#5-60	7-292								
RETURN	6-200	7-221	7-227	7-237	7-248	7-254	7-268	7-275	7-281	7-298
	7-310	7-328	7-342	7-348	7-354	7-360	7-372	7-379	7-387	7-394
	7-413	7-419	7-425	7-431	7-437	7-443	7-449	7-456	8-484	8-512
	9-543	9-559	9-566	9-573	9-579	10-654	11-662	11-667	11-672	11-677
	11-682	11-687	11-692	11-697	11-702	11-707	11-712	12-727	12-740	12-753
	12-766	12-779								
SAVRG	#5-60	7-289								
SLTDF\$	#5-60	5-62								
SOB	10-652									
SWSTK\$	10-597									
TTAB\$	#5-78	5-91	5-92	5-93	5-94	5-95	5-96	5-97	5-98	5-99
	5-100	5-101	5-102	5-103	5-104	5-105	5-106	5-107	5-108	5-109
	5-110	5-111	5-112	5-113	5-114	5-115	5-116	5-117	5-118	5-119
	5-120	5-121	5-122	5-123	5-124	5-125	5-126	5-127	5-128	5-129
	5-130	5-131	5-132	5-133	5-134	5-135	5-136	5-137	5-138	5-139
	5-140	5-141	5-142	5-143						

LOCAL DATA

```

521 001002
522 001004 012724 177777      101$: RETURN
523 001010                      MOV      #-1,(R4)+ ; UNW NOT IN EITHER MXTAB OR MPTAB FILE
524                                RETURN
525                                ; SINGLE WORD
526                                WORD:
527                                MOV      (R5)+,(R4)+ ; COPY A WORD
528 001012 112524              MOV      (R5)+,(R4)+
529 001014 112524              MOV      (R5)+,(R4)+
530 001016                      RETURN

```


ADDRB 000156R	DSSISK= 000000	LN.OPE= 000001	PRDPR 000512R	002 SLNB 000572R	
ADDRW 000202R	DSSL11= 000001	LN.REF= 000002	PRFLAG 000162R	002 SLNW 000616R	
APR 002142R	DSSYNC= 000000	LN.SER= 000002	PRI 001374R	STLIN 000164R	002
ASSCHK= 000000	DSSYNM= 000000	LN.STA= 000017	PRINC 000206R	002 STNB 000646R	
ASSCPS= 000000	EQF 000452R	LN.SUB= 000360	PRLINX 000220R	002 STNW 000704R	
ASSPRI= 000000	ESSXPR= 000000	LN.TRI= 000006	PRLSAD 001224R	002 SSSWRG= 000000	
ASSTPP= 000000	FILE 001414R	LSTHD 000462R	PRMTP 000350R	002 SSSYSZ= 007600	
BLKB 000232R	FLGS 000160R	002 LTAB 000500R	PRMUX 000514R	002 S.COST 000001	
BLKW 000244R	FL.DDM= ***** GX	LSSASG= 000000	PRNAME 000200R	002 S.FLG 000000	
BYTE 000256R	FL.DLC= ***** GX	LSSDRV= 000000	PRPAR 000210R	002 S.LEN 000004	
CORE 001020R	FL.KMX= ***** GX	LSSP11= 000001	PRPCB 000204R	002 S.NMST 000002	
CSR 000262R	FL.MUX= ***** GX	LSS11R= 000000	PRPCHA 001062R	002 S.OWNR 000003	
CTIM 000274R	FSSLLVL= 000001	L.COST 000015	PRPDVA 000176R	002 TABLE 000002R	002
CSSCKP= 000000	GSSTPP= 000000	L.CTL 000012	PRPRI 000344R	002 TEMP 000000R	002
CSSORE= 000400	GSSSTS= 000000	L.CVA 177776	PRSCSR 000516R	002 TIME 002110R	
CSSRSH= 177564	GSSTTK= 000000	L.DDM 000002	PRSLN 000260R	002 TSSKMG= 000000	
DECPY = ***** GX	GSSWRD= 000000	L.DDS 000004	PRSLTA 000252R	002 TSSMIN= 000000	
DPRB 002170R	ICBAD 000166R	002 L.DLC 000003	PRSNUM 000352R	002 UNB 000746R	
DPRW 002176R	ICBLN 000174R	002 L.DLM 000006	PRTBL 000216R	002 UNW 000770R	
DVCHA 000302R	ICBPT 000170R	002 L.DLS 000010	PRTOV 000566R	VSSCTR= 001000	
D\$AMXC 000072	INDEX 000156R	002 L.FLG 000000	PRVCT 000346R	002 WORD 001012R	
D\$AMXH 000074	INT 001222R	L.KRBA 000016	PR2HLD 001326R	002 XSSDBT= 000000	
D\$ANN 000000	ISSRAR= 000000	L.LEN = 000022	PR2MBS 001246R	002 X2CH01 002306R	
D\$BRPR 000102	ISSRDN= 000000	L.MPF 000022	PR2MRT 001266R	002 X2CH02 002334R	
D\$BRM 000100	KSARS = ***** GX	L.NMST 000020	PR2MWS 001226R	002 X2CH03 002362R	
D\$DELF 000045	KSSCNT= 177546	L.NSTA 000014	PR2RET 001306R	002 X2CH04 002410R	
D\$DELW 000046	KSSCSR= 177546	L.OWNR 000021	PR3CAL 000234R	002 X2CH05 002436R	
D\$END = 000104	KSSLDC= 000000	L.UNT 000013	PR3CLR 000236R	002 X3CH01 002204R	
D\$FNB 000034	KSSTPS= 000074	MPTAB 001474R	PR3DBS 000224R	002 X3CH02 001112R	
D\$HIOR 000024	LB.ADR= ***** GX	MXPTB 001754R	PR3DWS 000230R	002 X3CH03 002220R	
D\$HOST 000022	LD\$LP = 000000	MXTAB 001406R	PR3MBS 000226R	002 X3CH04 002226R	
D\$INAC 000044	LFILE 001502R	MSSCRB= 000124	PR3MCL 000244R	002 X3CH05 002234R	
D\$INCT 000042	LF.ACT= 100000	MSSCRX= 000000	PR3MRE 000246R	002 X3CH06 002242R	
D\$IPL 000051	LF.BRO= 000400	MSSFCS= 000000	PR3MRS 000250R	002 X3CH07 002250R	
D\$IID 000020	LF.BWT= 000007	MSSMGE= 000000	PR3MWS 000232R	002 X3CH08 002256R	
D\$LNAM 000006	LF.ENA= 002000	MSSNET= 000000	PR3RES 000240R	002 X3CH09 002264R	
D\$LNUM 000014	LF.LPB= 001000	MSSOVR= 000000	PR3RST 000242R	002 X3CH10 002272R	
D\$LST 000047	LF.MDC= 000100	NOD 000554R	PR7 = ***** GX	X3CH11 002300R	
D\$MAXC 000064	LF.MFL= 004000	NXTLN 000172R	002 PSSP45= 000000	ZTIME = ***** GX	
D\$MAXH 000066	LF.MTP= 000020	NSSACC= 000001	PSSWRD= 000000	\$ADDR = ***** GX	
D\$MAXV 000070	LF.PAC= 000200	NSSBUF= 000001	QSSOPT= 000010	\$BIAS = ***** GX	
D\$MLL 000040	LF.RDY= 040000	NSSLDV= 000001	RSSDER= 000000	\$BIN = ***** GX	
D\$MNOD 000041	LF.REA= 010000	NSSMCP= 000001	RSSK11= 000001	\$FLAGS= ***** GX	
D\$NA 000062	LF.SER= 000040	NSSMLL= 000001	RSSSND= 000000	\$ICBAD= ***** GX	
D\$NEA 000056	LF.TIM= 000010	NSSMOV= 000010	RSS11M= 000000	\$ICBLN= ***** GX	
D\$NBRA 000054	LF.UNL= 020000	NSSNCT= 000001	SAVALC 001170R	\$JSROF= ***** GX	
D\$NEND= 000054	LF.X2P= 000000	NSSPEM= 000001	SCOM 001132R	\$LDA = ***** GX	
D\$NLN 000030	LIBR 001652R	PAPR 000432R	002 SECSR 000506R	\$MAPX = ***** GX	
D\$NN 000060	LINKS 000454R	PECHA 000404R	SF.ACT= 000200	\$MXS12= ***** GX	
D\$OUTT 000043	LN.CLO= 000000	PRAPR 000154R	002 SF.ENA= 000100	\$PDVA = ***** GX	
D\$RETF 000050	LN.DUM= 000005	PRBIAS 000202R	002 SF.LPB= 000004	\$PTR = ***** GX	
D\$RNN 000002	LN.LOA= 000004	PRCSR 000342R	002 SF.MFL= 000040	\$SLTSV= ***** GX	
D\$RTMR 000076	LN.LDO= 000003	PRCTIM 000222R	002 SF.PAC= 000020	\$SMTSZ= ***** GX	
D\$SEG 000036	LN.OAU= 000003	PRDCHA 000556R	002 SF.REA= 000010	\$TMEX6 000000RG	
D\$SER 000032	LN.OFF= 000001	PRDDPV 000256R	002 SF.SER= 000001	\$UBIAS= ***** GX	
D\$SQRL 000052	LN.ON= 000000	PRDEV 000214R	002 SF.SVC= 000002	\$UDA = ***** GX	
D\$SUBG= 177514	LN.OOP= 000004	PRDLPV 000254R	002 SF.UNL= 000040	\$VECT1= ***** GX	

```

183      ;+
184      ; TMEX7 - EXPAND BINARY INFORMATION INTO REAL INFORMATION
185
186      ; THIS ROUTINE IS CALLED AFTER THE ASCII TEMPLATE FILE HAS BEEN READ
187      ; AND PROCESSED INTO THE BINARY FILE. THIS ROUTINE IS CALLED ONLY FOR
188      ; LINE TABLES LOCATED AT THE END OF THE PROCESS PARTITION.
189
190      ; INPUTS:
191      ; $BIN=START OF BINARY INFORMATION
192      ; $PTR=END OF BINARY INFORMATION
193      ; $BIAS=BIAS OF THE PROCESS PARTITION
194      ; $SLTA=SLT ADDRESS
195      ; $ADDR=ADDRESS OF REAL DATA
196
197      ; OUTPUTS:
198      ; ALL REGISTERS=DESTROYED
199
200      ; TMEX7::
201      000000      005067      000350'      CLR      INDEX      ; ZERO INDEX VALUE
202      000004      005067      000352'      CLR      FLAGS      ; AND SPECIAL FLAGS
203      000010      016767      000000G      000346'      MOV      $FLAGS,PRFLAG ; SAVE A LOCAL COPY OF $FLAGS
204      000016      012703      000020      MOV      #16.,R3      ; MOVE 16. SECONDARY CSR'S
205      000022      005005      000000G      000154'      5$:      CLR      R5      ; INITIALIZE INDEX
206      000024      016565      000000G      000154'      5$:      MOV      $$CSR(R5),PRCSR(R5) ; GET LOCAL COPY OF SECONDARY CSR'S
207      000032      062705      000002      ADD      #2,R5      ; ADJUST INDEX
208      000036      005303      DEC      R3      ; COPY 16.
209      000040      003371      BGT      5$
210      000042      016767      000000G      000214'      MOV      $$DCHA,PRDCHA ; GET LOCAL COPY OF DEVICE CHARACTERISTICS
211      000050      016767      000002G      000216'      MOV      $$DCHA+2,PRDCHA+2 ;
212      000056      016767      000000G      000224'      MOV      $$PCHA,PRPCHA ; GET LOCAL COPY OF PROTOCOL EMULATOR CHAR
213      000064      016767      000000G      000222'      MOV      $$PRI,PRPRI ; GET LOCAL COPY OF DEVICE PRIORITY
214      000072      012703      000010      MOV      #8.,R3      ; MOVE 8 WORDS
215      000076      005005      CLR      R5      ; INITIALIZE INDEX
216      000100      016565      000000G      000226'      501$:      MOV      $X2MWS(R5),PR2MWS(R5) ; SAVE LOCAL COPY OF MAX WINDOW SIZE
217      000106      062705      000002      ADD      #2,R5      ; ADJUST INDEX
218      000112      SOB      R3,501$
219      000116      012703      000010      MOV      #8.,R3      ; MOVE 8 WORDS
220      000122      005005      CLR      R5      ; INITIALIZE INDEX
221      000124      016565      000000G      000246'      502$:      MOV      $X2MBS(R5),PR2MBS(R5) ; SAVE LOCAL COPY OF MAX BLOCK SIZE
222      000132      062705      000002      ADD      #2,R5      ; ADJUST INDEX
223      000136      SOB      R3,502$
224      000142      012703      000010      MOV      #8.,R3      ; MOVE 8 WORDS
225      000146      005005      CLR      R5      ; INITIALIZE INDEX
226      000150      016565      000000G      000266'      503$:      MOV      $X2MRT(R5),PR2MRT(R5) ; SAVE LOCAL COPY OF MAX RETRANSMIT COUNT
227      000156      062705      000002      ADD      #2,R5      ; ADJUST INDEX
228      000162      SOB      R3,503$
229      000166      012703      000010      MOV      #8.,R3      ; MOVE 8 WORDS
230      000172      005005      CLR      R5      ; INITIALIZE INDEX
231      000174      016565      000000G      000306'      504$:      MOV      $X2RET(R5),PR2RET(R5) ; SAVE LOCAL COPY OF RETRANSMIT TIMER
232      000202      062705      000002      ADD      #2,R5      ; ADJUST INDEX
233      000206      SOB      R3,504$
234      000212      012703      000010      MOV      #8.,R3      ; MOVE 8 WORDS
235      000216      005005      CLR      R5      ; INITIALIZE INDEX
236      000220      016565      000000G      000326'      505$:      MOV      $X2HLD(R5),PR2HLD(R5) ; SAVE LOCAL COPY OF HOLDBACK TIMER
237      000226      062705      000002      ADD      #2,R5      ; ADJUST INDEX
238      000232      SOB      R3,505$
239      000236      016703      000000G      MOV      $ADDR,R3      ; GET LINE TABLE ADDRESS

```

```

X3CH
721          .SBTTL  X3CH
722
723          ;
724          ; DEFAULT BLOCK SIZE
725          ;
726          X3CH01: TST      (R3)+
727          001570 005723   RETURN
728          ;
729          ; MAXIMUM BLOCK SIZE
730          ;
731          X3CH02: TST      (R3)+
732          001576 005723   RETURN
733          ;
734          ; DEFAULT WINDOW SIZE
735          ;
736          X3CH03: INC      R3
737          001600 005203   RETURN
738          ;
739          ; MAXIMUM WINDOW SIZE
740          ;
741          X3CH04: INC      R3
742          001606 005203   RETURN
743          ;
744          ; CALL TIMER
745          ;
746          X3CH05: INC      R3
747          001610 005203   RETURN
748          ;
749          ; CLEAR TIMER
750          ;
751          X3CH06: INC      R3
752          001616 005203   RETURN
753          ;
754          ; RESET TIMER
755          ;
756          X3CH07: INC      R3
757          001620 005203   RETURN
758          ;
759          ; RESTART TIMER
760          ;
761          X3CH08: INC      R3
762          001624 005203   RETURN
763          ;
764          ; MAXIMUM CLEAR VALUE
765          ;
766          X3CH09: INC      R3
767          001630 005203   RETURN
768          ;
769          ; MAXIMUM RESETS VALUE
770          ;
771          X3CH10: INC      R3
772          001636 005203   RETURN
773          ;
774          ; MAXIMUM RESTARTS VALUE
775          ;
776          X3CH11: INC      R3
777          001640 005203   RETURN

```

```

105      ;+
106      $TM.IN - INITIALIZE TEMPLATE DATA, OPEN PRIMARY TEMPLATE
107
108      : INPUTS:
109      :     NONE
110
111      : OUTPUTS:
112      :     C-BIT=SUCCESS/FAILURE
113      :     $ADDR=LINE TABLE ADDRESS IF MUX UPDATE
114      :     R0,R1,R2=DESTROYED
115      : -
116      : .ENABL  LSB
117      $TM.IN::
118      000000 016767 000000G 000000G  MOV    $XBUF2,$BIN      ; COPY END OF TASK BUFFER ADDRESS
119      000000 016767 000000G 000000G  MOV    $XBUF2,$PTR      ; TWICE
120      000014 016767 000000G 000000G  MOV    $XLEN2,$LEFT     ; COPY BUFFER SIZE
121      000022 005067 000000G          CLR    $ADDR             ; CLEAR LINE TABLE ADDRESS
122      000026 005067 000000G          CLR    $MXADD            ; CLEAR MUX LINE TABLE ADDRESS
123      000032 005067 000000G          CLR    $MXSIZ            ; CLEAR MUX TABLE SIZE
124      000036 005067 000000G          CLR    $OFF              ; CLEAR LINE TABLE OFFSET
125      000042 005067 000000G          CLR    $SIZE             ; CLEAR LINE TABLE SIZE
126      000046 012767 000000G 000000G  MOV    #F1.UMR,$FLAG1    ; INITIALIZE LINE TABLE FLAGS
127      000054 005067 000000G          CLR    $NVECT            ; CLEAR NUMBER OF VECTORS
128      000060 005067 000000G          CLR    $NALL             ; CLEAR NUMBER OF ALLOCATIONS
129      000064 012767 000000G 000000G  MOV    #$$SZPTR,$SZPTR    ; INITIALIZE SIZE TABLE POINTER
130      000072 016700 000000G          MOV    $NAME,R0           ; GET THREE CHARACTER PDV NAME
131      000076          CALL    $DFN                             ; SETUP DEFAULT FILE NAME
132      000102 016701 000000G          MOV    $PDVA,R1           ; GET POINTER TO PDV
133      000106 032761 000000G 000000G  BIT    #ZF.PSE,Z.FLG(R1) ; IS THIS A PSEUDO DDM?
134      000114 001037          BNE     10$                       ; IF NE, YES ... IT HAS NO VECTORS
135      000116 032767 000000G 000000G  BIT    #FL.DDM,$FLAGS    ; IS THIS A DDM LINE TABLE ?
136      000124 001433          BEQ     10$                       ; NO ..
137      000126 016700 000000G          MOV    $SLTA,R0           ; GET THE SLT ADDRESS
138      000132 016067 000004 000000G  MOV    L.DDS(R0),$ADDR    ; HAS THE LINE TABLE ALREADY BEEN BUILT ?
139      000140 001425          BEQ     10$                       ; NO ..
140
141      000142 052767 000000G 000000G  BIS    #FL.CHA,$FLAGS    ; YES .. IT'S CHARACTERISTICS UPDATE
142      000150 016000 000016          MOV    L.KRBA(R0),R0      ; GET THE KRB ADDRESS
143
144      : IF DF R$$MPL
145      :     $CPU
146      :     BEQ 3$
147      :     MOV K.URM(R0),SETAF+S.AFAF ; GET THE UNIBUS RUN MASK
148      :     DIR$ #SETAF ; SET AFFINITY TO BUS
149
150      3$:
151      : .ENDC ; DF R$$MPL
152
153      000154 116001 000001          MOV    K.VCT(R0),R1      ; GET THE VECTOR ADDRESS
154      000160 006301          ASL     R1                       ; CONVERT TO REALL ADDRESS
155      000162 006301          ASL     R1
156      000164          CALL    FNDADD ; FIND LINE TABLE ADDRESS
157
158      : IF DF R$$MPL
159      :     $CPU
160      :     BEQ 4$
161      :     RMAF$$
162
163      4$:

```

TMPSTB - NTL TEMPLATE STATE TAB MACRO V05.03b Saturday 29-Jun-85 00:45 ^{6 9} Page 6
ENTRY POINT

62
63
64
65
66
67 000000
68 000000

.SBTTL ENTRY POINT
:
: ENTRY POINT CALLED FROM 'NTLTMP'
: \$TPARS::
: RETURN

TMPSTB - NTL TEMPLATE STATE TAB MACRO V05.03b Saturday 29-Jun-85 00:45 Page 16-2
Symbol table

```

ASSCHK= 000000      F2.END= ***** GX      NSSLDV= 000001      $BINF = ***** GX      $MPLHD= ***** GX
ASSCPS= 000000      F2.NEG= ***** GX      NSSMCP= 000001      $BLANK= 000006      $MPBTB = ***** GX
ASSPRI= 000000      F2.SUB= ***** GX      NSSMLL= 000001      $BLKB = ***** GX      $MXPTB= ***** GX
ASSTRP= 000000      G$STPP= 000000      NSSMOV= 000010      $BLKW = ***** GX      $MXTAB= ***** GX
BIN      000500R      002 G$STSS= 000000      NSSNCT= 000001      $BYTE = ***** GX      $NOD = ***** GX
BIN1     000516R      002 G$STTK= 000000      NSSPEM= 000001      $CFILE= ***** GX      $NOMOD= ***** GX
BIN2     000546R      002 G$SWRD= 000000      ONLY1  001712R      002 $CNB = ***** GX      $NUM = ***** GX
BLKB     000554R      002 INT  001072R      002 ONLY2  001726R      002 $CNW = ***** GX      $NUMBR= 000002
BLKW     000572R      002 I$SRAR= 000000      OPERAT 000034R      002 $CORE = ***** GX      $ODD = ***** GX
BYTE     000610R      002 I$SRDN= 000000      PERIOD 000024R      002 $CORE1= ***** GX      $PECHA= ***** GX
BYTE1    000634R      002 K$SCNT= 177546      POSEXP 001606R      002 $CORE2= ***** GX      $PRI = ***** GX
BYTE2    000644R      002 K$SCSR= 177546      PRI     001350R      002 $CSR = ***** GX      $PRI1 = ***** GX
CORE     000672R      002 K$SLDC= 000000      PRTOV   001374R      002 $CTM = ***** GX      $PROV = ***** GX
CORE1    000710R      002 K$STPS= 000074      P$P45= 000000      002 $DIGIT= 000024      $RAD50= 000016
CORE2    000740R      002 LD$LP = 000000      P$SWRD= 000000      002 $DNUMB= 000014      $SCOM = ***** GX
CSR       000746R      002 LFILE 001174R      002 Q$SDPT= 000010      002 $DPRB = ***** GX      $SECSR= ***** GX
C$SCKP= 000000      002 LFILE1 001210R      002 R$SDER= 000000      002 $DPRW = ***** GX      $SLNB = ***** GX
C$SORE= 000400      002 LFILE2 001226R      002 R$SK11= 000001      002 $DVCHA= ***** GX      $SLNW = ***** GX
C$SRSH= 177564      002 LIBR  001110R      002 R$SND= 000000      002 $END1 = ***** GX      $STNB = ***** GX
D$BBUG= 177514      002 LIBR1  001136R      002 R$S11M= 000000      002 $END2 = ***** GX      $STNW = ***** GX
D$BISK= 000000      002 LIBR2  001166R      002 SCOM   001332R      002 $EOS = 000012      $STRNG= 000004
D$SL11= 000001      002 L$ASG= 000000      002 SYMBOL 001630R      002 $EVEN = ***** GX      $SUBXP= 000010
D$SYNC= 000000      002 L$SDRV= 000000      002 S$SWRG= 000000      002 $EXIT = 000000      $TIME = ***** GX
D$SYNM= 000000      002 L$SP11= 000001      002 S$SYZ= 007600      002 $FAIL = 177777      $TMPKW 000000RG      003
END       001760R      002 L$S11R= 000000      002 T$KMG= 000000      002 $FILE = ***** GX      $TMPST 000000RG      002
EOF       000756R      002 MOD    001660R      002 T$SMIN= 000000      002 $FLAG2= ***** GX      $TPARS 000000RG
EOF1     001000R      002 MODIFY 001636R      002 UMR    001412R      002 $FNAME= *** GX      $UMROF= ***** GX
EOF2     001010R      002 MPTAB  001236R      002 VFY    001434R      002 $GLC  = * **** GX      $UMRON= ***** GX
ERROR    001742R      002 MPTAB1 001252R      002 V$SCTR= 001000      002 $ = ***** GX      $UNB = ***** GX
ERR$1 = ***** GX      002 MXPTB  001306R      002 WORD   001452R      002 $ = ***** GX      $UNW = ***** GX
ERR$3 = ***** GX      002 MXPTB1 001322R      002 WORD1  001476R      002 $INT1 = ***** GX      $WORD = ***** GX
ERR$4 = ***** GX      002 MXTAB  001262R      002 WORD2  001506R      002 $INT2 = ***** GX      $X2CHB= ***** GX
ERR$5 = ***** GX      002 MXTAB1 001276R      002 X$SDBT= 000000      002 $INT3 = * * * GX      $X2CHW= ***** GX
ERR$6 = ***** GX      002 M$SCRB= 000124      002 X2CHB  001544R      002 $LAMD= C 00      $X3CHB= ***** GX
ERR$7 = ***** GX      002 M$SCRX= 000000      002 X2CHW  001564R      002 $LFILE= * * GX      $X3CHW= ***** GX
EXP       001574R      002 M$SFCS= 000000      002 X3CHB  001534R      002 $LFLHD= * * * GX      $ZBYTE= ***** GX
E$SXPR= 000000      002 M$SMGE= 000000      002 X3CHW  001554R      002 $LIBR = ***** GX      $ZWORD= ***** GX
FILE      001026R      002 M$SNET= 000000      002 X$CDRB= ***** GX      $LIBR1= ***** GX
FILE1     001042R      002 M$SOVR= 000000      002 $ADDRB= ***** GX      $LINKS= ***** GX
FNAME     001666R      002 NEG    001622R      002 $ADDRW= ***** GX      $LSTD= ***** GX
FNAME1    001670R      002 NONE   001676R      002 $ALPHA= 000022      002 $LTAB = ***** GX      $MOD = ***** GX
F$SLVL= 000001      002 N$SACC= 000001      002 $ANY = 000020      002 $MOD = ***** GX
F2.BYT= ***** GX      002 N$SBUF= 000001      002 $APR = ***** GX
. ABS.    000000      000 (RW,I,GBL,ABS,OVR)
          000002      001 (RW,I,LCL,REL,CON)
$STATE    001770      002 (RW,D,LCL,REL,CON)
$KTAB     000162      003 (RW,D,LCL,REL,CON)
$KSTR     000435      004 (RW,D,LCL,REL,CON)
Errors detected: 0

```

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 11288 Words (45 Pages)

```

380 000446 016703 000000G      MOV      SLTMA,R3      ; GET SLT TABLE ADDRESS
381 000452 062703 000000G      ADD      #AB,R3      ;
382 000456 011303      MOV      (R3),R3      ; VECTORED EXEC ACCESS
383
384 000460 012302      20$:    MOV      (R3)+,R2      ; GET NEXT SLT
385 000462 126062 000002 000002  CMPB     L,DDM(R0),L,DDM(R2) ; DO DDM PDV INDICES MATCH?
386 000470 001016      BNE      40$      ; IF NE, NO
387 000472 126062 000012 000012  CMPB     L,CTL(R0),L,CTL(R2) ; DO CONTROLLER NUMBERS MATCH?
388 000500 001012      BNE      40$      ; IF NE, NO
389 000502 010162 000004      MOV      R1,L,DDS(R2) ; SET DDM LINE TABLE ADDRESS
390 000506 126262 000002 000003  CMPB     L,DDM(R2),L,DLC(R2) ; COMBINATION DDM/DLC?
391 000514 001002      BNE      30$      ; IF NE, NO
392 000516 010162 000010      MOV      R1,L,DLS(R2) ; SET DLC LINE TABLE ADDRESS
393
394 000522 166701 000000G      30$:    SUB      $MXLEN,R1      ; GET NEXT UNIT PORTION ADDRESS
395 000526 005316      40$:    DEC      (SP)      ; ANY MORE ENTRIES LEFT ?
396 000530 003353      BGT      20$      ; IF GT, YES
397 000532 005726      TST      (SP)+      ; CLEAN OFF THE STACK
398 000534 000410      BR       70$      ; DON'T SET DDM LINE TABLE ADDRESS AGAIN !
399 000536 010160 000004      50$:    MOV      R1,L,DDS(R0) ; SET DDM DATA BASE ADDRESS
400
401
402 000542 126060 000002 000003 55$:    CMPB     L,DDM(R0),L,DLC(R0) ; COMBINATION DDM/DLC?
403 000550 001002      BNE      70$      ; IF NE, NO
404 000552 010160 000010      60$:    MOV      R1,L,DLS(R0) ; SET DLC DATA BASE ADDRESS
405
406 000556      70$:    CALL     $TMRDY      ; MARK LINE READY IF APPROPRIATE
407 000562      80$:    RETURN

```

UNLHC - UNLOAD HOP COST MATRI MACRO V05.03b Saturday 29-Jun-85 00:48 Page 6-1

DEAMHC - DEALLOCATE MINHOP/COST,CKT CTRS.,ROL

112	000236	066301	000030	ADD	N\$ROA2(R3),R1	:	...
113	000242	006301		ASL	R1	:	CONVERT TO BYTES
114	000244			ASR\$	6,R1	:	CONVERT TO BLOCKS
115	000260	016300	000024	MOV	N\$ROA1+2(R3),R0	:	GET APR BIAS
116	000264	001402		BEQ	25\$:	IF NONE - BRANCH
117	000266			CALL	\$DEA18	:	DEALLOCATE BLOCK
118							
119	000272			25\$:	RETURN		
120							
121		000001			.END		

UNLPSI - NTL UNLOAD ROUTINES FO MACRO V05.03b Wednesday 17-Jul-85 12:05 Page 9-1
XLDTE - DEALLOCATE LOCAL DTE BLOCK

```
241 000422          CALL    $XDEAC      ;; DEALLOCATE BLOCK
242 000426 000714    BR      10$        ;; DO FOR NEXT BLOCK
243                                     ;;
244 000430          40$: RESMAP          ;; RESTORE PREVIOUS MAPPING
245 000434          50$: RETURN
246                                     ;;
247                                     .ENDC ; NDF I$SAS
```

UNLPSI - NTL UNLOAD ROUTINES FO MACRO V05.03b Wednesday 17-Jul-85 12:05 Page 10
XLDTE - DEALLOCATE PORT TABLE AND CIRCUIT BLOCKS

```

68 LOCAL DATA
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88 000000
89 000002

        .SBTTL LOCAL DATA
        ;
        ; LOCAL DATA
        ;
        .IF DF R$$MPL
        ;
        ; DEFINE CTB SYMBOL BLOCK
        ;
        CTBLK: .WORD 0,0
        CTBSYM: .RAD50 '$CTXXX'
        CTVAL: .BLKW 1
        CETNAM: .ASCIZ /CETAB/
        STBEXT: .ASCIZ /STB/
                .EVEN
        .ENDC

        ;
        ; READY LINES ON THIS CONTROLLER
        ;
        UNITS: .BLKW 1
                .PSECT

```

6 15

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

.TITLE XLBR - RELEASE LIBRARY DESCRIPTOR
.IDENT /V05.00/

.. COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

.. THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

.. THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

.. DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

.. MODULE DESCRIPTION:

.. NTL - LIBRARY DESCRIPTOR DE-ALLOCATION ROUTINE

.. DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

.. IDENT HISTORY:

- .. 1.00 27-FEB-78
.. VERSION 2.0 RELEASE
- .. 2.00 14-DEC-79
.. DECNET-11M/S V3.0
.. DECNET-11M-PLUS V1.0
- .. 3.00 16-APR-82
.. DECNET-11M V3.1
.. DECNET-11M-PLUS V1.1
- .. 4.00 07-NOV-83
.. DECNET-11M V4.0
.. DECNET-11M-PLUS V2.0
- .. 5.00 22-JUL-85
.. DECnet-11M/S V4.2
.. DECnet-11M-Plus V3.0
.. DECnet-Micro/Rsx V1.0

H 15

XPAR CREATED BY MACRO ON 29-JUN-85 AT 00:49 PAGE 2 G 16
MACRO CROSS REFERENCE CREF 04.00
MACRO NAME REFERENCES

CALL	6-76	6-110
PCBDF\$	#5-57	5-58
RETURN	6-113	
SWSTK\$	6-76	

TMPEX3 CREATED BY MACRO ON 29-JUN-85 AT 00:43 PAGE 1 H 1

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
M\$SMGE	= 000000	8-205 8-250 9-265
NULL	000050 R	5-80 5-81 5-87
		5-100 5-103 5-104
		5-117 5-118 5-119
		5-126 5-127 5-128
POP1	000044 R	5-82 5-83 5-84
		#7-181 5-91 5-113
POP2	000040 R	5-91 5-110 5-114
POP3	000032 R	5-94 5-97 #7-169
P.REL	= ***** GX	8-212
P.WAIT	= ***** GX	8-217
R\$S11M	= 000000	8-204
TABLE	000000 R	#5-80 6-158
XCORE	000172 R	5-85 #8-246
XLBR	000224 R	5-92 #9-264
XXPTB	000252 R	5-115 #10-284
XSCOM	000052 R	5-102 #8-196
\$BIN	= ***** GX	6-151
\$DEAC2	= ***** GX	8-219
\$DEA16	= ***** GX	8-240 10-295
\$DEA18	= ***** GX	8-238
\$MAPX	= ***** GX	8-213
\$MXLEN	= ***** GX	10-292
\$PCB	= ***** GX	8-208
\$PRAVL	= ***** GX	8-216 8-220
\$PTR	= ***** GX	6-152
\$TMEX3	000000 RG	#6-150
\$XLBR	= ***** GX	9-279

```

458
459      ; SECONDARY FILE
460
461 000524 005267 000154' MXTAB:  VC  FLAGS      ; INDICATE MUX EXPANSION
462 000530 112546      FILE:  MOVB  (R5)+, -(SP)  ; GET REPEAT COUNT
463 000532 010546      10$:  MOV   R5, -(SP)      ; SAVE CURRENT BINARY POINTER
464 000534 011605      TST   FLAGS             ; RESTORE BINARY POINTER
465 000536 005767 000154' BEQ   20$           ; IS IT MUX EXPANSION ?
466 000542 001407      CMPB  L, UNT(R4), INDEX ; NO ..
467 000544 126467 000013 000152' BNE  20$      ; DO UNIT NUMBERS MATCH ?
468 000552 001003      CMP   L, DDS(R4), R3    ; NO ..
469 000554 026403 000004      BNE  101$        ; IS THE LINE TABLE POINTER CORRECT ?
470 000560 001025      CLR   R2               ; NO .. ERROR
471 000562 005002      20$:  CLRB (R5)+, R2     ; GET FUNCTION CODE
472 000564 152502      CMPB  #, EOF, R2        ; IS IT END OF FILE?
473 000566 122702 000000G BEQ   30$          ; IF EQ, YES
474 000572 001404      ASLB  R2               ; CONVERT TO A WORD INDEX
475 000574 106302      CALL  @TABLE-2(R2)      ; EXPAND A FIELD
476 000576      BR      20$                  ; KEEP LOOKING FOR EOF
477 000602 000767      INC   INDEX            ; TALLY ONE PASS THRU SECONDARY FILE
478 000604 005267 000152' 30$:  CMPB  2(SP), INDEX ; ARE WE DONE?
479 000610 126667 000002 000152' BNE  10$      ; IF NE, NO
480 000616 001346      CMP   (SP)+, (SP)+      ; PURGE STACK
481 000620 022626 000152' CLR   INDEX          ; CLEAR SECONDARY INDEX
482 000622 005067 000154' CLR   FLAGS         ; AND SPECIAL FLAGS
483 000626 005067 000154' RETURN
484 000632
485
486      ; ERROR CONDITION
487
488
489 000634 101$:  CRASH      ; LINE TABLE POINTER (R3) IS SCREWED UP
490
491
492      ; LINKED SECONDARY FILE
493
494 000636      MPTAB:  CRASH
495 000640 112546      LFILE:  MOVB  (R5)+, -(SP)  ; SAVE REPEAT COUNT
496 000642 122525      CMPB  (R5)+, (R5)+      ; DISCARD LISTHEAD DISPLACEMENT
497 000644 010546      MOV   R5, -(SP)      ; SAVE CURRENT BINARY POINTER
498 000646 011605      10$:  MOV   (SP), R5     ; RESET BINARY POINTER
499 000650 005002      20$:  CLRB R2          ; GET FUNCTION CODE
500 000652 152502      CMPB  (R5)+, R2        ; IS IT END OF FILE?
501 000654 022702 000000G BEQ   30$          ; IF EQ, YES
502 000660 001404      ASLB  R2               ; CONVERT TO WORD INDEX
503 000662 106302      CALL  @TABLE-2(R2)      ; EXPAND A FIELD
504 000664      BR      20$                  ; KEEP LOOKING FOR EOF
505 000670 000767      INC   INDEX            ; TALLY ONE PASS THRU SECONDARY FILE
506 000672 005267 000152' 30$:  CMPB  2(SP), INDEX ; ARE WE DONE?
507 000676 126667 000002 000152' BNE  10$      ; IF NE, NO
508 000704 001360      CMP   (SP)+, (SP)+      ; PURGE STACK
509 000706 022626 000152' CLR   INDEX          ; CLEAR SECONDARY INDEX
510 000710 005067 000154' CLR   FLAGS         ; AND SPECIAL FLAGS
511 000714 005067 000154' RETURN
512 000720

```

FILEID**TMPEX6

```

TTTTTTTTTT  MM      MM  PPPPPPP  EEEEEEEEE  XX      XX  666666
TTTTTTTTTT  MM      MM  PPPPPPP  EEEEEEEEE  XX      XX  666666
TT          MMMM  MMMM  PP          EE          XX      XX  66
TT          MMMM  MMMM  PP          EE          XX      XX  66
TT          MM    MM    PP          EE          XX  XX  66
TT          MM    MM    PP          EE          XX  XX  66
TT          MM    MM    PPPPPPP  EEEEEEEEE  XX      XX  66666666
TT          MM    MM    PPPPPPP  EEEEEEEEE  XX      XX  66666666
TT          MM    MM    PP          EE          XX  XX  66  66
TT          MM    MM    PP          EE          XX  XX  66  66
TT          MM    MM    PP          EE          XX  XX  66  66
TT          MM    MM    PP          EEEEEEEEE  XX      XX  666666
TT          MM    MM    PP          EEEEEEEEE  XX      XX  666666

```

```

....
....
....
....

```

```

LL          SSSSSSSS  TTTTTTTTTT
LL          SSSSSSSS  TTTTTTTTTT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SSSSSS    TT
LL          SSSSSS    TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LLLLLLLLLLLL  SSSSSSSS  TT
LLLLLLLLLLLL  SSSSSSSS  TT

```

```

532
533
534
535 001020 112524
536 001022 112524
537 001024 112524
538 001026 112524
539 001030 112546
540 001032 112566 000001
541 001036 012603
542 001040 001433
543
544 001042 011146
545 001044 016446 177774
546 001050 016711 000000G
547 001054 016700 000000G
548
549 001060 011620
550 001062 006303
551 001064 005203
552 001066 010320
553 001070 010067 000000G
554
555 001074 012611
556 001076 105765 177771
557 001102 100011
558 001104 010200
559 001106 005303
560 001110
561 001120 005020
562 001122
563
564 001126 012611
565
566 001130

; POINTER TO ALLOCATED CORE
CORE:  MOVB  (R5)+,(R4)+ ; COPY FIRST WORD OF ADDRESS
        MOVB  (R5)+,(R4)+ ; COPY SECOND WORD OF ADDRESS
        MCVB  (R5)+,(R4)+ ; ASSEMBLE COUNT
        MOVB  (R5)+,-(SP) ; ...
        MOV   (R5)+,1(SP) ; IF ZERO, NO UNLOAD DATA TO STORE
        BEQ   30$
        MOV   (R1),-(SP) ; SAVE CURRENT MAPPING
        MOV   -4(R4),-(SP) ; SAVE FIRST WORD OF ADDRESS (BIAS)
        MOV   $UBIAS,(R1) ; GET UNLOAD BLOCK BIAS
        MOV   $UDA,R0 ; GET UNLOAD DATA ADDRESS
        MOV   (SP),(R0)+ ; STORE ALLOCATION BIAS
        ASL   R3 ; SHIFT COUNT LEFT 1 PLACE
        INC   R3 ; SET BIT 0 (ADDRESS=BIAS, NOT OFFSET)
        MOV   R3,(R0)+ ; STORE COUNT
        MOV   R0,$UDA ; SET NEW UNLOAD DATA ADDRESS
        MOV   (SP)+,(R1) ; MAP TO THE CORE ALLOCATION
        TSTB  -7(R5) ; IS THIS .CORE OR .BIN?
        BPL   20$ ; IF PL, .BIN
        MOV   R2,R0 ; MAP TO THE .CORE ALLOCATION
        DEC   R3 ; CONVERT TO A WORD COUNT
        ASL$   4,R3
        CLR   (R0)+ ; ZERO THE ALLOCATION
        SOB   R3,10$ ; ...
        MOV   (SP)+,(R1) ; RESTORE MAPPING
        RETURN
10$:
20$:
30$:

```


TMPEX6 - NTL TEMPLATE EXPANSION MACRO V05.03b Saturday 29-Jun-85 ^{H 5}00:43 Page 16-2
Symbol table

.EOF = ***** GX

. ABS.	177776	000	(RW,I,GBL,ABS,OVR)
	002464	001	(RW,I,LCL,REL,CON)
DATA	001346	002	(RW,D,LCL,REL,CON)

Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 12325 Words (49 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:22.43
SY:TMPEX6.V2,[132,134]TMPEX6/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]TMPEX6

```

MACRO DEFINITIONS

240 000242 016705 000000G      MOV    $SLTA,R5      ; GET THE SLT ADDRESS
241 000246 116504 000002      MOV    L.DDM(R5),R4    ; GET THE DDM PDV INDEX
242 000252                      CALL   FLT16          ; SEARCH THRU THE UNLOAD BLOCKS
243                                ; FOR THE START OF THE LINE TABLE
244 000256 010504              MOV    R5,R4          ; COPY SLT ADDRESS
245 000260 005767 000356'      TST    UBA           ; SUCCESS ?
246 000264 001434              BEQ    101$          ; IF EQ, NO
247
248                                .IF DF R$$11M
249 000266                      SWSTK$ 20$          ; * ENTER KERNEL MODE
250                                .ENDC
251
252 000272 016700 000000G      MOV    $BIAS,R0        ; * GET BIAS FOR PARTITION LINE TABLE
253 000276                      CALL   $MAPX          ; * MAP TO LINE TABLE
254 000302 016705 000000G      MOV    $BIN,R5        ; * START OF BINARY DATA
255 000306 016703 000356'      MOV    UBA,R3         ; * GET UNLOAD BLOCK ADDRESS
256 000312 042703 177700      BIC    #<77>,R3       ; * USE LOW 6 BITS ONLY
257 000316 016303 000360'      MOV    BLOCK(R3),R3   ; * GET CONTROLLER TABLE STARTING ADDRESS
258 000322 042703 160000      BIC    #160000,R3     ; * USE OUR OWN MAPPING REGISTER
259 000326 050203              BIS    R2,R3         ; *
260 000330 026705 000000G      10$: CMP    $PTR,R5   ; * IS IT END OF BINARY DATA?
261 000334 001407              BEQ    20$           ; * IF EQ, YES
262 000336 005002              CLR    R2            ; * GET FUNCTION CODE
263 000340 152502              BISB   (R5)+,R2      ; *
264 000342 001404              BEQ    20$           ; * IF ZERO, END OF FILE
265 000344 106302              ASLB   R2            ; * CONVERT TO A WORD INDEX
266 000346                      CALL   @TABLE-2(R2) ; * EXPAND ONE FIELD
267 000352 000766              BR     10$           ; * KEEP LOOPING UNTIL END
268 000354                      20$:
269                                .IF DF R$$11D ! I$SAS
270                                CALL   @($P)+      ; * CALL CO-ROUTINE TO RESTORE MAPPING
271                                .ENDC
272
273 000354                      RETURN                ; * BACK TO USER MODE AND THEN TO CALLER
274
275                                ;
276                                ; ERROR CONDITION
277                                ;
278 000356                      101$: CRASH          ; CAN'T FIND LINE TABLE IN UNLOAD BLOCKS

```

```

779          .SBTTL X2CH
780          ;
781          ; MAXIMUM WINDOW SIZE
782          ;
783          X2CH01: TST  FLAGS          ; IS IT MUX EXPANSION ?
784          BEQ    10$              ; NO
785          MOV    INDEX,R0          ; GET SECONDARY FILE COUNTER
786          CMPB   L,UNT(R4),R0      ; DO UNIT NUMBERS MATCH ?
787          BNE    10$              ; NO
788          ASL    R0                ; CONVERT TO A WORD INDEX
789          MOVB   PR2MWS(R0),(R3)+  ; STORE MAXIMUM WINDOW SIZE
790          BR     20$              ; AND EXIT
791          10$: INC    R3
792          20$: RETURN
793          ;
794          ; MAXIMUM BLOCK SIZE
795          ;
796          X2CH02: TST  FLAGS          ; IS IT MUX EXPANSION ?
797          BEQ    10$              ; NO
798          MOV    INDEX,R0          ; GET SECONDARY FILE COUNTER
799          CMPB   L,UNT(R4),R0      ; DO UNIT NUMBERS MATCH ?
800          BNE    10$              ; NO
801          ASL    R0                ; CONVERT TO A WORD INDEX
802          MOVB   PR2MBS(R0),(R3)+  ; STORE MAXIMUM BLOCK SIZE
803          BR     20$              ; AND EXIT
804          10$: TST  (R3)+
805          20$: RETURN
806          ;
807          ; MAXIMUM RETRANSMIT COUNT
808          ;
809          X2CH03: TST  FLAGS          ; IS IT MUX EXPANSION ?
810          BEQ    10$              ; NO
811          MOV    INDEX,R0          ; GET SECONDARY FILE COUNTER
812          CMPB   L,UNT(R4),R0      ; DO UNIT NUMBERS MATCH ?
813          BNE    10$              ; NO
814          ASL    R0                ; CONVERT TO A WORD INDEX
815          MOVB   PR2MRT(R0),(R3)+  ; STORE MAXIMUM RETRANSMIT COUNT
816          BR     20$              ; AND EXIT
817          10$: INC    R3
818          20$: RETURN
819          ;
820          ; RETRANSMIT TIMER
821          ;
822          X2CH04: TST  FLAGS          ; IS IT MUX EXPANSION ?
823          BEQ    10$              ; NO
824          MOV    INDEX,R0          ; GET SECONDARY FILE COUNTER
825          CMPB   L,UNT(R4),R0      ; DO UNIT NUMBERS MATCH ?
826          BNE    10$              ; NO
827          ASL    R0                ; CONVERT TO A WORD INDEX
828          MOVB   PR2RET(R0),(R3)+  ; STORE RETRANSMIT TIMER
829          BR     20$              ; AND EXIT
830          10$: TST  (R3)+
831          20$: RETURN
832          ;
833          ; HOLDBACK TIMER
834          ;
835          X2CH05: TST  FLAGS          ; IS IT MUX EXPANSION ?

```

```

162                                     .ENDC      ; DF R$$MPL
163
164 000170 020167 000000G      CMP      R1,$$VECT      ; IS THIS THE SAME VECTOR ADDRESS ?
165 000174 001004      BNE      5$      ; IF NE, NO .. WARN USER
166 000176 026760 000000G 000002      CMP      $$CSR,K.CSR(R0)      ; IS THIS THE SAME CSR ADDRESS ?
167 000204 001403      BEQ      10$      ; IF EQ, YES .. OKAY
168 000206      5$:      EMMSG$      WZ      ; WARNING - MUX UPDATE ONLY
169 000214 005000      10$:      CLR      R0      ; SETUP ASCII FILE SPEC
170 000216 005001      CLR      R1      ; FOR [13],5X]XXX.DAT
171 000220 012702 000000'      MOV      #TMPEXT,R2      ; ...
172 000224      CALL      $AZFS      ; ...
173 000230      CALL      $CLQUE      ; TRY TO OPEN PRIMARY TEMPLATE
174 000234 105777 000000G      TSTB      @SCLIOS      ; SUCCESS? (CLEARS C-BIT)
175 000240 100401      BMI      101$      ; IF MI, NO
176 000242      RTSPC:      RETURN
177
178      ;
179      ; ERROR CONDITIONS
180
181 000244 032767 000000G 000000G 101$:      BIT      #FL,LLC,$FLAGS      ; IS THIS AN LLC?
182 000252 001405      BEQ      102$      ; IF EQ, NO
183 000254 016700 000000G      MOV      $PDVA,R0      ; GET PDV ADDRESS
184 000260 012760 177777 000000G      MOV      #-1,Z.DAT(R0)      ; MARK DATA BASE AS EMPTY
185 000266      102$:      EMMSG$R      8      ; OPEN FAILURE
186      .DSABL      LSB

```

```

70          .SBTTL STATE TABLES
71
72          ***
73          : STATE TABLES
74          ***
75          000002          ISTAT$ $TMPST,$TMPKW
76
77          :
78          : START OF OPERATOR ('.'), LABEL, OR END OF STRING
79          :
80          000002          STATES$
81          000002          TRANS$ !END,$EXIT
82          000002          TRANS$ !,OPERATOR
83          000002          TRANS$ $RAD50
84
85          :
86          : COLON FOLLOWING LABEL
87          :
88          000002          STATES$
89          000002          TRANS$ ':
90
91          :
92          : OPTIONAL SECOND COLON FOLLOWING LABEL
93          :
94          000002          STATES$
95          000002          TRANS$ ':,PERIOD
96          000002          TRANS$ $LAMDA
97
98          :
99          : START OF OPERATOR OR END OF STRING
100         :
101         000002          STATES$ PERIOD
102         000002          TRANS$ !END,$EXIT
103         000002          TRANS$ '
104

```

TMPSTB - NIL TEMPLATE STATE TAB MACRO V05.03b Saturday 29-Jun-85 ^{H 10} 00:45 Page 16-3
Symbol table

Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:02:34.07
SY:TMPSTB.V2,[132,134]TMPSTB/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]TMPSTB

```

409
410
411
412
413
414
415
416
417
418
419 000564 016701 000000G $TMRDY::MOV $DLC PV,R1 ; GET DLC PDV ADDRESS
420 000570 001403 BEQ 10$ ; IF EQ, COMBINATION DDM/DLC
421 000572 005761 000000G TST Z.PCB(R1) ; IS DLC CORE-RESIDENT?
422 000576 001420 BEQ 20$ ; IF EQ, NO
423 000600 016701 000000G 10$: MOV $DDMPV,R1 ; GET DDM PDV ADDRESS
424 000604 005761 000000G TST Z.PCB(R1) ; IS DDM CORE-RESIDENT?
425 000610 001413 BEQ 20$ ; IF EQ, NO
426
427 000612 005760 000004 TST L.DDS(R0) ; IS DDM LINE TABLE PRESENT?
428 000616 001410 BEQ 20$ ; IF EQ, NO
429 000620 005760 000010 TST L.DLS(R0) ; IS DLC LINE TABLE PRESENT?
430 000624 001405 BEQ 20$ ; IF EQ, NO
431 000626 CALL MANINI ; INITIALIZE MANAGEMENT FIELDS
432 000632 052760 040000 000000 BIS #LF.RDY,L.FLG(R0) ; MARK LINE AS READY
433
434 .IF DF R$MPL
435
436 MOV L.KRBA(R0),R1 ; GET THE KRB ADDRESS
437 BIT #F1.VCT,$FLAG1 ; DID WE SETUP THE VECTORS ?
438 BEQ 20$ ; IF EQ, NO .. LEAVE THE DEVICE OFFLINE
439 TSTB @NCPU ; IS THIS A MULTIPROCESSOR SYSTEM ?
440 BEQ 15$ ; IF EQ, NO .. SKIP URM CHECK
441 BIT K.URM(R1),@URMST ; IS THIS BUS ON-LINE ?
442 BEQ 20$ ; IF EQ, NO
443 15$: BIC #KS.OFL,K.STS(R1) ; WHY DIDN'T YOU SAY SO ?
444
445 .ENDC
446
447 000640 20$: RETURN

```

AT\$ACL= 000100	D\$LNAM 000006	E\$NNOD 000002	N\$LV2 000010	T\$LIF 000013
AT\$ADN= 000040	D\$LNUM 000014	E\$NRT 000042	N\$MHC1 000036	T\$LIFL 000013
AT\$AUP= 000020	D\$LST 000047	E\$NRT 000005	N\$MHC2 000044	T\$LIFO 000013
AT\$CYC= 000004	D\$MAXC 000064	E\$NSEG 000010	N\$PLD 000016	T\$LIFS 000013
AT\$LV1= 000002	D\$MAXH 000066	E\$NTIM 000046	N\$PRI 000076	T\$LIN 000000
AT\$LV2= 000001	D\$MAXV 000070	E\$NUSE 000004	N\$ROA1 000022	T\$LIPS 000006
AT\$NEX= 000200	D\$MLL 000040	E\$STRT 000006	N\$ROA2 000030	T\$LLD 000012
AT\$UP = 000010	D\$MNOD 000041	F\$XPR= 000000	N\$RTMX 000014	T\$LLDC 000045
ASCIR 000003	D\$NA 000062	F\$LV1= 000001	N\$RTM1 000014	T\$LLDL 000012
ASLEN 000004	D\$NBEA 000056	G\$TTP= 000000	N\$RTM2 000015	T\$LLDO 000012
ASNID 000000	D\$NBRA 000054	G\$TSS= 000000	N\$RT1 000000	T\$LLDS 000012
ASTM 000003	D\$NEND= 000054	G\$TTK= 000000	N\$RT2 000006	T\$LLEN 000046
ASTMI 000002	D\$NLN 000030	G\$WRD= 000000	N\$TCTL 000112	T\$LOPR 000002
ASTSZ 000000	D\$NN 000060	I\$RAR= 000000	N\$TLC 000100	T\$LTCL 000024
ASTYP 000002	D\$OUTT 000043	I\$RDN= 000000	N\$TNC 000114	T\$LTIM 000026
A\$CHK= 000000	D\$RETF 000050	K\$CNT= 177546	N\$TRC 000106	T\$LTPR 000014
A\$CPS= 000000	D\$RNN 000002	K\$CSR= 177546	N\$TTCB 000110	T\$LTPS 000020
A\$PRI= 000000	D\$RTMR 000076	K\$DC= 000000	N\$VER 000066	T\$NAPL 000004
A\$TRP= 000000	D\$SEG 000036	K\$TPS= 000074	N\$XLEN 000124	T\$NFE 000000
C\$CKP= 000000	D\$SER 000032	LD\$LP = 000000	N\$ACC= 000001	T\$NLEN 000010
C\$DRE= 000400	D\$SQL 000052	L\$ASG= 000000	N\$BUF= 000001	T\$NNUL 000002
C\$RSH= 177564	D\$BUG= 177514	L\$DRV= 000000	N\$LDV= 000001	T\$NOPL 000006
DECPT = ***** GX	D\$ISK= 000000	L\$P11= 000001	N\$MCP= 000001	T\$NRNI 000042
D\$AMXC 000072	D\$L11= 000001	L\$11R= 000000	N\$ML= 000001	T\$NRPL 000005
D\$AMXH 000074	D\$YNC= 000000	M\$CRB= 000124	N\$MOV= 000010	T\$NRUL 000007
D\$ANN 000000	D\$YNM= 000000	M\$CRX= 000000	N\$NCT= 000001	T\$NVR 000001
D\$BRPR 000102	E\$NBR 000014	M\$FCS= 000000	N\$PEM= 000001	T\$RPRI 000040
D\$BRIM 000100	E\$NBS 000020	M\$MGE= 000000	N\$P45= 000000	T\$SYC 000034
D\$DELF 000045	E\$NCR 000034	M\$NET= 000000	P\$WRD= 000000	T\$T5 000030
D\$DELW 000046	E\$NCS 000036	M\$OVR= 000000	Q\$OPT= 000010	T\$T6 000032
D\$END = 000104	E\$NIC 000044	N\$ADJ1 000072	R\$DER= 000000	T\$KMG= 000000
D\$FNB 000034	E\$NLEN 000050	N\$ADJ2 000074	R\$K11= 000001	T\$MIN= 000000
D\$HIOR 000024	E\$NLLA 000012	N\$CACH 000062	R\$SND= 000000	V\$CTR= 001000
D\$HOST 000022	E\$NLNK 000000	N\$CRC 000120	R\$11M= 000000	X\$DBT= 000000
D\$INAC 000044	E\$NML 000040	N\$HC1 000052	S\$WRG= 000000	\$DEAHC 000000RG
D\$INCT 000042	E\$NMR 000024	N\$HC2 000056	S\$YSZ= 007600	\$DEA18= ***** GX
D\$IPL 000051	E\$NMS 000030	N\$LV1 000002	T\$FLAG 000044	\$MUL = ***** GX
D\$LIO 000020				

. ABS. 000124 000 (RW,I,GBL,ABS,OVR)
 000274 001 (RW,I,LCL,REL,CON)
 Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 13258 Words (52 Pages)
 Size of core pool: 14440 Words (55 Pages)
 Operating system: RSX-11M/PLUS

Elapsed time: 00:00:09.49
 SY:UNLHC.V2,[132,134]UNLHC/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]UNLHC


```

249                                     .SBTTL XPORT - DEALLOCATE PORT TABLE AND CIRCUIT BLOCKS
250
251                                     *
252                                     XPORT - DEALLOCATE PORT TABLE AND ASSOCIATED X.25 CIRCUIT BLOCKS
253                                     INPUTS:
254                                     NONE
255                                     OUTPUTS:
256                                     THE PORT TABLE AND THE X.25 CIRCUIT BLOCK IS DEALLOCATED
257                                     -
258
259                                     .IF NDF ISSAS
260                                     XPORT: MOV @PSIPT,R3 ; GET ADDRESS OF PSI HOME BLOCK
261                                     MOV H$PTB(R3),R0 ; GET ADDRESS OF PORT TABLE
262                                     MOV H$NPT(R3),R4 ; GET NUMBER OF PORTS
263
264                                     SWSTK$ 30$ ; ENTER SYSTEM STATE
265                                     SAVMAP ; SAVE MAPPING
266                                     MOV R0,-(SP) ; GET UNMAPPED ADDRESS OF PORT TABLE
267                                     CALL $CEACX ; MAP TO PORT TABLE
268                                     MOV (SP)+,R2 ; RETRIEVE MAPPED PORT TABLE ADDRESS
269                                     TST (R2)+ ; SKIP FIRST ENTRY
270                                     DEC R4 ;
271                                     BLE 25$ ; IF NO CIRCUIT BLOCKS - BRANCH
272
273                                     10$: MOV (R2)+,R0 ; GET ADDRESS OF X.25 CIRCUIT BLOCK
274                                     BEQ 20$ ; BR IF NONE FOR THIS PORT
275                                     MOV #X$LEN,R1 ; GET LENGTH OF CIRCUIT BLOCK
276                                     SAVMAP ; SAVE PORT TABLE MAPPING
277                                     SAVRG <R2> ; AND ADDRESS
278                                     CALL $XDEAC ; DEALLOCATE CIRCUIT BLOCK
279                                     RESRG <R2> ; RESTORE PORT TABLE ADDRESS
280                                     RESMAP ; AND MAPPING
281                                     DEC R4 ; MORE TO CHECK?
282                                     BGT 10$ ; BR IF YES
283                                     25$: MOV R$R0+2(SP),R0 ; GET UNMAPPED ADDRESS OF PORT TABLE
284                                     MOV R$R4+2(SP),R1 ; GET NUMBER OF PORTS
285                                     ASL R1 ; MULTIPLY BY 2 TO OBTAIN BYTE COUNT
286                                     CALL $XDEAC ; DEALLOCATE PORT TABLE
287                                     RESMAP ; RESTORE MAPPING
288                                     30$: RETURN
289
290                                     .ENDC
291
292
293

```

91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121

000002
000002 010546
000004 017746 000000G
000010 017705 000000G
000014 022504
000016 001403
000020 005316
000022 003374
000024
000026 005726
000030 167705 000000G
000034 005745
000036 010504
000040 012605
000042

.SBTTL GETIND - GET PDV INDEX

```

: *
: GETIND - GET PDV INDEX FROM PDV ADDRESS
:
: INPUTS:
:   R4 = PDV ADDRESS
:
: OUTPUTS:
:   R4 = PDV INDEX
:   ALL OTHER REGISTERS PRESERVED
: -

```

```

GETIND::
MOV     R5, -(SP)           ; SAVE R5
MOV     @PDVNM, -(SP)      ; GET # OF PDV ENTRIES
MOV     @PDVTA, R5         ; POINT TO PDV MAPPING TABLE

10$:    CMP     (R5)+, R4    ; POINTING AT OUR ENTRY ?
        BEQ     20$         ; IF EQ, YES
        DEC     (SP)        ; ELSE, ONE LESS PDV TO COUNT
        BGT     10$         ; IF GT, MORE TO GO
        CRASH   10$         ; ELSE, CRASH - CAN'T FIND PDV ENTRY

20$:    TST     (SP)+       ; CLEAN UP STACK
        SUB     @PDVTA, R5  ; CALCULATE INDEX

        TST     -(R5)       ;
        MOV     R5, R4      ; RETURN INDEX IN R4
        MOV     (SP)+, R5   ; RESTORE R5
        RETURN

```

```

55                                     ;***
56                                     ; LOCAL DATA
57                                     ;***
58                                     .PSECT DATA,D
59 000000
60
61                                     ;
62                                     ; KERNEL MODE STORAGE (MAPPED ONLY)
63                                     ;
64                                     .IF DF M$$MGE
65 000000 LBADR: .BLKW 1 ; DESCRIPTOR VIRTUAL ADDRESS
66 000002 LBALL: .BLKW 1 ; LIBRARY BIAS WITHIN NTPool
67 000004 LBSIZ=LBADR ; LIBRARY SIZE IN 32. WORD BLOCKS
68 000004 LBIAS: .BLKW 1 ; LIBRARY BLOCK BIAS IF DE-ALLOCATION NEEDED
69 000006 NLBLK: .BLKW 1 ; NETLDR DATA BLOCK BIAS
70                                     .ENDC
71
72                                     ;
73                                     ; 32. WORD BLOCK ALLOCATION SIZE
74                                     ;
75 000001 .IIF DF M$$MGE, BLKSIZ=1
76 .IIF NDF M$$MGE, BLKSIZ=100
77
78 000000 .PSECT

```

FILEID**XXBUF

```

XX      XX  XX      XX  BBBB BBBB  UU      UU  FFFFFFFF
XX      XX  XX      XX  BBBB BBBB  UU      UU  FFFFFFFF
XX      XX  XX      XX  BB      BB  UU      UU  FF
XX      XX  XX      XX  BB      BB  UU      UU  FF
XX      XX  XX      XX  BB      BB  UU      UU  FF
XX      XX  XX      XX  BB      BB  UU      UU  FF
XX      XX  XX      XX  BBBB BBBB  UU      UU  FFFFFFFF
XX      XX  XX      XX  BBBB BBBB  UU      UU  FFFFFFFF
XX      XX  XX      XX  BB      BB  UU      UU  FF
XX      XX  XX      XX  BB      BB  UU      UU  FF
XX      XX  XX      XX  BB      BB  UU      UU  FF
XX      XX  XX      XX  BB      BB  UU      UU  FF
XX      XX  XX      XX  BBBB BBBB  UU      UU  FFFFFFFF
XX      XX  XX      XX  BBBB BBBB  UU      UU  FFFFFFFF

```

```

LL      SSSSSSSS  TTTTTTTTTT
LL      SSSSSSSS  TTTTTTTTTT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SSSSSS  TT
LL      SSSSSS  TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SSSSSSSS  TT
LL      SSSSSSSS  TT

```

TMPEX3 CREATED BY MACRO ON 29-JUN-85 AT 00:43 PAGE 2 I 1

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
CALL	6-158 8-209 8-213 8-219 8-240 8-258 10-295
CALLR	9-279
RETURN	6-160 7-170 7-176 7-182 7-187 8-241 8-259 10-296
SWSTK\$	8-209
TTAB\$	#5-67 5-80 5-81 5-82 5-83 5-84 5-85 5-86 5-87 5-88
	5-89 5-90 5-91 5-92 5-93 5-94 5-95 5-96 5-97 5-98
	5-99 5-100 5-101 5-102 5-103 5-104 5-105 5-106 5-107 5-108
	5-109 5-110 5-111 5-112 5-113 5-114 5-115 5-116 5-117 5-118
	5-119 5-120 5-121 5-122 5-123 5-124 5-125 5-126 5-127 5-128
	5-129 5-130 5-131 5-132

```

514
515      ; SECONDARY LINKED FILE
516
517      000722 005267 000154'  MXPTB: INC  FLAGS      ; INDICATE MUX EXPANSION
518      000726 005723          TST   (R3)+      ; POINT PAST FIRST LINE TABLE ADDRESS
519      000730 010346          MOV   R3,-(SP)    ; SAVE OUTPUT FILE POINTER
520      000732 112546          MOVB  (R5)+,-(SP) ; GET REPEAT COUNT
521      000734 122525          CMPB  (R5)+,(R5)+ ; SKIP ALLOCATION ADDRESS
522      000736 016303 177776  MOV   -2(P3),R3  ; GET FIRST LINE TABLE ADDRESS
523      000742 010546          MOV   R5,-(SP)    ; SAVE CURRENT BINARY FILE POINTER
524      000744 011605          MOV   (SP),R5    ; RESTORE BINARY FILE POINTER
525      000746 126467 000013 000152' 10$: CMPB  L,UNT(R4),INDEX ; DO UNIT NUMBERS MATCH ?
526      000754 001003          BNE   20$        ; IF NE, NO
527      000756 026403 000004          CMP   L,DDS(R4),R3 ; IS THE LINE TABLE POINTER CORRECT ?
528      000762 001026          BNE   101$       ; IF NE, NO .. ERROR
529      000764 005002          CLR    R2        ; GET FUNCTION CODE
530      000766 152502          BISB  (R5)+,R2    ;
531      000770 122702          CMPB  #.EOF,R2    ; IS IT END OF FILE ?
532      000774 001404          BEQ   30$        ; IF EQ, YES
533      000776 106302          ASLB  R2         ; CONVERT TO A WORD INDEX
534      001000          CALL  @TABLE-2(R2)       ; EXPAND A FIELD
535      001004 000767          BR     20$        ; GO TILL END OF ILE
536      001006 005267 000152' 30$: INC  INDEX   ; TALLY ONE PASS THRU SECONDARY FILE
537      001012 126667 000002 000152' CMPB  2(SP),INDEX ; ARE WE DONE ?
538      001020 001351          BNE   10$        ; IF NE, NO
539      001022 022626          CMP   (SP)+,(SP)+ ; PURGE STACK
540      001024 012603          MOV   (SP)+,R3    ; RESTORE OUTPUT FILE POINTER
541      001026 005067 000152' CLR  INDEX   ; CLEAR SECONDARY INDEX
542      001032 005067 000154' CLR  FLAGS    ; AND SPECIAL FLAGS
543      001036          RETURN
544
545      ; ERROR CONDITION
546
547      001040 101$: CRASH      ; LINE TABLE POINTER (R3) IS SCREWED UP
548
549
550      ; CURRENT TIME
551
552      001042 016746 000000G TIME: MOV   ZTIME,-(SP) ; GET $ZTIME ADDRESS
553      001044 062716 000002      ADD   #2,(SP) ; GET $ZTIME+2 ADDRESS
554      001052 017616 000000      MOV   @ (SP),(SP) ; GET $ZTIME+2 CONTENTS
555      001056 111623          MOVB  (SP),(R3)+ ; STORE TIME
556      001058 116623 000001      MOVB  1(SP),(R3)+
557      001060 005067          TST   (SP)+
558      001064 005726          RETURN
559      001066
560
561
562      ; ACTIVE POLLING RATIO (APR) - BYTE
563
564      001070 005203 APR: INC  R3
565      001072          RETURN
566
567
568      ; DEAD POLLING RATIO (DPRB) - BYTE
569
570

```

:LAA3.01
:LAA3.01
:LAA3.01
:LAA3.01
:**4

TMPEX6 - NTL TEMPLATE EXPANSION MACRO V05.03b Saturday 29-Jun-85 1 3 00:43
Table of contents

5-	51	MACRO DEFINITIONS
6-	80	LOCAL DATA
15-	880	X3CH
16-	938	X2CH

```

568
569
570
571
572 001132 112524
573 001134 112524
574 001136 112546
575 001140 112566 000001
576 001144 012603
577 001146 001424
578 001150 016400 177776
579 001154 010346
580 001156 006203
581 001160 005020
582 001162
583 001166 012603
584 001170 011146
585 001172 016446 177776
586
587 001176 016711 000000G
588 001202 016700 000000G
589 001206 012620
590 001210 010320
591 001212 010067 000000G
592 001216 012611
593 001220
594

;
; POINTER TO ALLOCATED SCOM
;
SCOM: .ENABL LSB
      MOV (R5)+,(R4)+ ; COPY ADDRESS OF SCOM ALLOCATION
      MOV (R5)+,(R4)+
      MOV (R5)+,-(SP) ; ASSEMBLE COUNT OF ALLOCATED
      MOV (R5)+,1(SP) ; BYTES, WHICH MUST ALWAYS
      MOV (SP)+,R3 ; BE EVEN (BIT 0 = 0)
      BEQ 20$ ; IF ZERO, NO UNLOAD DATA TO STORE
      MOV -2(R4),R0 ; GET SCOM ADDRESS
      MOV R3,-(SP) ; SAVE BYTE COUNT
      ASR R3 ; CONVERT TO A WORD COUNT
      CLR (R0)+ ; ZERO THE ALLOCATED SCOM
      SOB R3,10$
      MOV (SP)+,R3 ; RESTORE THE BYTE COUNT
      MOV (R1)+,-(SP) ; SAVE THE CURRENT MAPPING
      MOV -2(R4)+,-(SP) ; SAVE THE ALLOCATION ADDRESS
      MOV $UBIAS,(R1) ; MAP TO THE UNLOAD BLOCK
      MOV $UDA,R0 ; GET UNLOAD DATA ADDRESS
      MOV (SP)+,(R0)+ ; STORE ALLOCATION ADDRESS
      MOV R3,(R0)+ ; STORE COUNT
      MOV R0,$UDA ; SET NEW UNLOAD DATA ADDRESS
      MOV (SP)+,(R1) ; RESTORE OLD MAPPING
      RETURN
      .DSABL LSB
10$:
20$:

```


SYMBOL	VALUE	REFERENCES								
ADDRB	000156 R	6-94	#8-300							
ADDRW	000202 R	6-95	#8-310							
APR	0002142 R	6-125	#14-859							
BLKB	000232 R	6-96	#8-321							
BLKW	000244 R	6-97	#8-330							
BYTE	000256 R	6-98	#8-339							
CORE	001020 R	6-99	#9-535							
CSR	000262 R	6-100	#8-345							
CTIM	000274 R	6-101	#8-352							
DECP	= ***** GX	8-443								
OPRB	002170 R	6-126	#14-871							
OPRW	002176 R	6-127	#14-877							
DVCHA	000302 R	6-102	#8-358							
D\$NUM	000014	8-444								
EOF	000452 R	6-103	#8-399							
FILE	001414 R	6-104	#12-717							
FLAGS	000160 R	#6-170	*7-260	8-363	8-388	8-426	8-457	8-468	8-480	8-493
		8-508	*12-716	*12-732	*12-738	*12-772	*14-812	*14-840	14-859	
FL.DDM	= ***** GX	8-300	8-310	8-359	8-384					
FL.DLC	= ***** GX	8-300	8-310							
FL.KMX	= ***** GX	8-429								
FL.MUX	= ***** GX	8-361	8-386	16-943	16-953	16-963	16-973	16-983		
ICBAD	000166 R	#6-181	*7-263							
ICBLN	000174 R	#6-184	*7-264	11-703						
ICBPT	000170 R	#6-182	*7-262	11-618	11-702	*11-703				
INDEX	000156 R	#6-165	*7-259	8-365	8-390	8-428	8-459	8-470	8-484	8-497
		8-510	*12-727	*12-728	*12-731	*12-767	*12-768	*12-771	*14-831	
		14-832	*14-839	14-861	16-945	16-955	16-965	16-975	16-985	
INT	001222 R	6-105	#11-599							
IS\$AS	= *****	9-560								
KSARS	= ***** GX	11-600	11-603	11-654	11-669					
LB.ADR	= ***** GX	13-795								
LFILE	001502 R	6-108	#12-739							
LF.ACT	= 100000	#5-58								
LF.BRO	= 000400	#5-58								
LF.BWT	= 000007	#5-58								
LF.ENA	= 002000	#5-58								
LF.LPB	= 001000	#5-58								
LF.MDC	= 000100	#5-58								
LF.MFL	= 004000	#5-58								
LF.MTP	= 000020	#5-58								
LF.PAC	= 000200	#5-58								
LF.RDY	= 040000	#5-58								
LF.REA	= 010000	#5-58								
LF.SER	= 000040	#5-58								
LF.TIM	= 000010	#5-58								
LF.UNL	= 020000	#5-58								
LF.X2P	= 000000	#5-58								
LIBR	001652 R	6-106	#13-778							
LINKS	000454 R	6-107	#8-404							
LN.CLO	= 000000	#5-58								
LN.DUM	= 000005	#5-58								

```

280
281
282
283
284
285
286
287
288
289
290
291
292
293 000360 005203
294 000362
295
296
297
298
299 000364 005723
300 000366
301
302
303
304
305 000370 005000
306 000372 152500
307 000374 001002
308 000376 012700 000400
309 000402 060003
310 000404
311
312
313
314
315 000406 005000
316 000410 152500
317 000412 001002
318 000414 012700 000400
319 000420 006300
320 000422 060003
321 000424
322
323
324
325
326 000426 122523
327 000430
328
329
330
331
332 000432
333 000432 062705 000006
334 000436 022323
335 000440
336

***
EXPANSION ROUTINES
WITH THE EXCEPTION OF "DVCHA", "PECHA" AND "SECSR", ALL PROCESSING
CONSISTS OF UPDATING THE BINARY BUFFER POINTER (R5) AND THE LINE TABLE
POINTER (R3).
***

; LOCAL PHYSICAL STATION ADDRESS (BYTE)
ADDRB: INC R3
RETURN

; LOCAL PHYSICAL STATION ADDRESS (WORD)
ADDRW: TST (R3)+
RETURN

; ZERO BYTE(S)
BLKB: CLR R0 ; GET REPEAT COUNT
      BISB (R5)+,R0 ;
      BNE 10$ ; ZERO BECOMES 256.
      MOV #256.,R0 ;
10$: ADD R0,R3
      RETURN

; ZERO WORD(S)
BLKW: CLR R0 ; GET REPEAT COUNT
      BISB (R5)+,R0 ;
      BNE 10$ ; ZERO BECOMES 256.
      MOV #256.,R0 ;
10$: ASL R0 ; 2 BYTES PER WORD
      ADD R0,R3
      RETURN

; SINGLE BYTE
BYTE: CMPB (R5)+,(R3)+
      RETURN

; POINTER TO ALLOCATED CORE
CORE: ADD #6,R5
      CMP (R3)+,(R3)+
      RETURN

```

```

836 002030 001411      BEQ      10$      ; NO
837 002032 016700 000350' MOV     INDEX,R0      ; GET SECONDARY FILE COUNTER
838 002036 126400 000013 CMPB    L,UNT(R4),R0    ; DO UNIT NUMBERS MATCH ?
839 002042 001004      BNE     10$      ; NO
840 002044 006300      ASL     R0        ; CONVERT TO A WORD INDEX
841 002046 016023 000326' MOV     PR2HLD(R0),(R3)+    ; STORE HOLDBACK TIMER
842 002052 000401      BR      20$      ; AND EXIT
843 002054 005723      10$: TST     (R3)+
844 002056      20$: RETURN
845
846      000001      .END

```

```

188
189
190
191
192
193
194
195
196
197
198
199
200
201 C00274
202
203
204
205
206
207 000274 011102
208
209 000276 010267 000000G
210 000302 066702 000000G
211 000306 162702 000004
212 000312 022712 004537
213 000316 001405
214 000320 022722 000137
215
216
217
218
219
220 000324 001003
221
222 000326 011267 000000G
223 000332
224
225
226
227
228
229
230 000332
231
232
233
234
235
236
237
238
239
240
241
242
243
244 000334

;+
; FNDADD - FIND LINE TABLE ADDRESS
;
; INPUTS:
; R1 = VECTOR ADDRESS
;
; OUTPUTS:
; $ADDR = LINE TABLE ADDRESS
; R1 = PRESERVED
; R2 = DESTROYED
;-

FNDADD:
    .IF DF M$$MGE
    .IF DF R$$MPL
    SWSTK$ 20$
    .IFTF ; DF R$$MPL
    ; * ENTER KERNEL MODE
    MOV (R1),R2
    ; * POINT AT ICB OR LINE TABLE
    MOV R2,$ADDR
    ; * ASSUME VECTOR POINTS AT LINE TABLE
    ADD $ICBLN,R2
    ; * POINT TO KEY WORD IN ICB
    SUB #4,R2
    ; *
    CMP #4537,(R2)
    ; * IF "JSR R5,X", THEN LINE TABLE ADDRESS
    BEQ 10$
    ; * IF "JMP LINE-TABLE", THEN LINE TABLE
    CMP #137,(R2)+
    .IFTF ; DF R$$MPL
    BEQ 5$
    ; * ..ADDRESS IS NEXT WORD
    CLR R2
    ; * ELSE, ERROR
    BR 10$
    ; * ...
    .IFF ; DF R$$MPL
    BNE 101$
    ; * ..ADDRESS IS NEXT WORD, ELSE ERROR
    .IFTF ; DF R$$MPL
    5$: MOV (R2),$ADDR
    ; * SAVE REAL LINE TABLE ADDRESS
    10$: MOV R2,6(SP)
    ; * RETURN LINE TABLE ADDRESS IN R2
    RETURN
    ; * BACK TO USER MODE
    20$: TST R2
    ; SUCCESS ?
    BEQ 101$
    ; IF EQ, NO
    .ENDC ; DF R$$MPL
    RETURN
    .IFF ; DF M$$MGE
    MOV (R1),R2
    ; GET ADDRESS OF LINE TABLE
    CMP #4537,(R2)
    ; MUST BE "JSR R5,X"
    BNE 101$
    ; IF NE, ERROR
    MOV R2,$ADDR
    ; SAVE ADDRESS OF LINE TABLE
    RETURN
    .ENDC ; DF M$$MGE
;
; ERROR CONDITIONS
;
; 101$: CRASH
; WHAT IS R2 POINTING AT ???

```

.SBTTL OPERATOR KEYWORDS STATE TABLES

```

106
107
108
109
110
111 000002      ; OPERATOR KEYWORDS
112 000002      ;
113 000002      STATES$ OPERATOR
114 000002      TRANS$ 'ADDRB',NONE,$ADDRB
115 000002      TRANS$ 'BLKB',BLKB
116 000002      TRANS$ 'BYTE',BYTE,F2.BYT,$FLAG2
117 000002      TRANS$ 'CNB',NONE,$CNB
118 000002      TRANS$ 'END',EOF,F2.END,$FLAG2
119 000002      TRANS$ 'FILE',FILE
120 000002      TRANS$ 'ODD',NONE,$ODD
121 000002      TRANS$ 'SLNB',NONE,$SLNB
122 000002      TRANS$ 'STNB',NONE,$STNB
123 000002      TRANS$ 'UMR',UMR
124 000002      TRANS$ 'UNB',NONE,$UNB
125 000002      TRANS$ 'VFX',VFX
126 000002      TRANS$ 'APR',NONE,$APR
127 000002      TRANS$ 'DPRB',NONE,$DPRB
128 000002      TRANS$ 'X3CHB',X3CHB
129 000002      TRANS$ 'X2CHB',X2CHB
130 000002      TRANS$ $LAMDA,$SEVEN
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

TMPSTB CREATED BY MACRO ON 29-JUN-85 AT 00:46 PAGE 1 I 10
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
M\$MGE	= 000000	9-186 10-238 12-305 14-386 16-506
\$ALPHA	= 000022	#7-75
\$ANY	= 000020	#7-75
\$BLANK	= 000006	#7-75
\$DIGIT	= 000024	#7-75
\$DNUMB	= 000014	#7-75
\$EOS	= 000012	#7-75
\$EXIT	= 000000	#7-75
\$FAIL	= 177777	#7-75
\$GPRM	= *****	7-75
\$LAMDA	= 000000	#7-75
\$NUMBR	= 000002	#7-75
\$RAD50	= 000016	#7-75
\$RONLY	= *****	7-75 7-75
\$STRNG	= 000004	#7-75
\$SUBXP	= 000010	#7-75
\$TMPKW	000000 RG	#7-75
\$TMPST	000000 RG	#7-75
\$TPARS	000000 RG	#6-67
\$\$\$FLG	= 177777	#7-75
\$\$\$KEY	= 177777	#7-75

```

449
450      *** - MANINI - INITIALIZE MANAGEMENT FIELDS OF SLT
451
452      INPUTS:
453          R0 = SLT ADDRESS
454
455      OUTPUTS:
456          R1,R2 = DESTROYED
457
458      -
459
460      MANINI:
461          MOV     @SLTNM,R2      ; GET NUMBER OF LINES IN SYSTEM
462          MOV     @SLTMA,R1      ; POINT TO SLT MAPPING TABLE
463
464      10$:  CMP     (R1)+,R0      ; FIND SLT POINTER IN TABLE
465           BEQ     20$           ; IF EQ, GOT IT
466           SOB     R2,10$       ; ELSE, KEEP LOOKING
467           CRASH    ; $$$?@ - NOT FOUND ??
468
469      20$:  TST     -(R1)         ; BACK UP
470           SUB     @SLTMA,R1     ; CALCULATE SLN*2
471           ADD     @LLCTA,R1     ; POINT AT ENTRY IN LLC TABLE
472           MOVB    L,NSTA(R0),R2 ; GET NUMBER OF TRIBUTARIES
473           BEQ     50$           ; IF EQ, NONE .. PT-TO-PT
474
475      ; MULTI-POINT LINE
476
477          MOV     R2,-(SP)       ; SAVE NUMBER OF TRIB ON STACK
478          MOV     R0,R2         ; POINT TO START OF TRIBUTARIES
479          ADD     #L.MPF,R2     ;
480          MOV     (R1),R1       ; GET ADDR/2 OF MULTI-POINT TABLE
481          BMI     30$           ; MAKE SURE SIGN BIT IS SET !!
482          CRASH    ; ELSE, CONSISTENCY ERROR
483          ASL     R1            ; FORM REAL ADDRESS
484
485      30$:  ASL     R1
486
487      40$:  TSTB    (R1)+         ; SKIP LLN
488           MOVB    (R1)+,S.OWNR(R2) ; FILL IN OWNER
489           MOVB    #LN.OFF,S.NMST(R2) ; AND INITIAL STATE IS OFF
490           ADD     #S.LEN,R2     ; NEXT TRIBUTARY
491           DEC     (SP)          ; ANY MORE ?
492           BGT     40$           ; IF GT, YES
493           TST     (SP)+         ; ELSE, CLEAN UP STACK
494           BR      60$          ; AND RETURN
495
496      ; PT-TO-PT LINE
497
498      50$:  MOVB    1(R1),L.OWNR(R0) ; FILL IN OWNER
499           MOVB    #LN.OFF,L.NMST(R0) ; AND INITIAL STATE IS OFF
500
501      60$:  RETURN
502
503      .END

```

SYMBOL	VALUE	REFERENCES
A\$LEN	000004	6-77
DECP	= ***** GX	6-73
D\$NBEA	000056	6-76
D\$NBRA	000054	6-75
D\$NLN	000030	6-74
E\$NBR	000014	#5-52
E\$NBS	000020	#5-52
E\$NCR	000034	#5-52
E\$NCS	000036	#5-52
E\$NIC	000044	#5-52
E\$NLEN	000050	#5-52
E\$NLLA	000012	#5-52
E\$NLNK	000000	#5-52
E\$NML	000040	#5-52
E\$NMR	000024	#5-52
E\$NMS	000030	#5-52
E\$NNOD	000002	#5-52
E\$NRT	000042	#5-52
E\$NRTP	000005	#5-52
E\$NSEG	000010	#5-52
E\$NTIM	000046	#5-52
E\$NUSE	000004	#5-52
E\$STRT	000006	#5-52
I\$SAS	= *****	6-81 6-96 6-106 6-114
N\$ADJ1	000072	6-71 6-82
N\$ADJ2	000074	6-87
N\$MHC1	000036	6-91 6-97
N\$MHC2	000044	6-92
N\$ROA1	000022	6-111 6-115
N\$ROA2	000030	6-112
N\$TLC	000100	6-102 6-107
R\$SEIS	= *****	6-81 6-96 6-106 6-114
R\$S11D	= *****	6-81 6-96 6-106 6-114
T\$FLAG	000044	#5-52
T\$LIF	000013	#5-52
T\$LIFL	000013	#5-52
T\$LIFO	000013	#5-52
T\$LIFS	000013	#5-52
T\$LIN	000000	#5-52
T\$LIPS	000006	#5-52
T\$LLD	000012	#5-52
T\$LLDC	000045	#5-52
T\$LLDL	000012	#5-52
T\$LLDO	000012	#5-52
T\$LLDS	000012	#5-52
T\$LLEN	000046	#5-52 6-101
T\$LOPR	000002	#5-52
T\$LTCL	000024	#5-52
T\$LTIM	000026	#5-52
T\$LTPR	000014	#5-52
T\$LTPS	000020	#5-52
T\$NAPL	000004	#5-52

295
 296
 297
 298
 299
 300
 301
 302
 303
 304
 305
 306
 307
 308
 309
 310
 311
 312
 313
 314
 315
 316
 317
 318

000560 017701 000000G
 000564 016100 000040
 000570 001406
 000572 005061 000040
 000576 012701 000016
 000602
 000606

```
.SBTTL XX29 - DEALLOCATE X.29 DATA BLOCK

;+
; XX29 - DEALLOCATE X.29 DATA BLOCK
;
; INPUTS
;   NONE
;
; OUTPUTS
;   R0, R1, R2 DESTROYED
;   THE X.29 DATA BLOCK IS DEALLOCATED
;-

      .IF NDF I$$$AS

XX29:  MOV    @PSIPT,R1      ; POINT TO HOME BLOCK
        MOV    H$X29C(R1),R0 ; GET X.29 BLOCK ADDRESS
        BEQ    10$          ; IF EQ, NOTHING TO DO
        CLR    H$X29C(R1)   ; FORGET POINTER
        MOV    #Q$LEN,R1    ; GET SIZE OF BLOCK
        CALL   $DEAT6       ; DEALLOCATE IT
10$:   RETURN                ; AND RETURN

      .ENDC
```

123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161

```
.SBTTL FNCTB - FIND CTB ADDRESS

+
FNCTB - FIND CTB ADDRESS
INPUT:
R4 = DDM PDV ADDRESS
OUTPUT:
R1 = CTB ADDRESS
R0,R2=DESTROYED
-
.IF DF R$$MPL
FNCTB::
MOV Z.NAM(R4),CTBSYM+2 ; STORE 2ND HALF OF NAME "$CTXXX"
MOV #CXUIC,R0 ; GET THE COMM EXEC UIC
MOV #CETNAM,R1 ; FORM FILE SPEC "LXX,5XCETAB.STB"
MOV #STBEXT,R2 ; ...
CALL $AZFS ; ...
MOV #CTBLK,R0 ; POINT AT SYMBOL BLOCK
CALL $SYMBL ; LOOK FOR SYMBOL IN STB FILE
BCS 101$ ; IF CS, ERROR
TST CTBLK+2 ; SYMBOL DEFINED ?
BEQ 111$ ; IF EQ, NO

SWSTK$ 10$ ; ENTER KERNEL MODE
MOV @NTLPT,R0 ; GET NTL HOME BLOCK BIAS
CALL $MAPX ; MAP TO NTL HOME BLOCK
MOV .CXALL(R2),R1 ; GET START ADDRESS OF CETAB
ADD CTVAL,R1 ; CALCULATE ADDRESS OF CTB
MOV R1,6(SP) ; RETURN CTB ADDRESS IN USER R1
; REMEMBER - MAPX ADDS ONE WORD TO SP
10$: RETURN ; BACK TO USER MODE AND TO CALLER
; ERROR CONDITIONS
101$: CRASH ; NEVER SHOULD HAVE GOT HERE !!!
111$: CRASH ; CETAB FORGOT TO DEFINE SYMBOL ?
.ENDC
```

```

80
81
82
83
84
85
86
87
88
89
90 000000
91 000000 042702 177700
92 000004 010267 000000'
93 000010 005067 000002'
94 000014 005067 000004'
95 000020 017767 000000G 000006'
96
97 000026
98
99 000032
100 000036 056702 000000'
101 000042 105362 000000G
102 000046 001041
103 000050 005012
104 000052 016267 000000G 000002'
105 000060 016267 000000G 000000'
106 000066 042702 000077
107 000072 026700 000006'
108
109 000076 001425
110 000100 005362 000002
111 000104 001022
112 000106 011246
113
114 000110 010067 000004'
115 000114 016711 000006'
116
117 000120 026200 000000G
118 000124 001003
119 000126 012662 000000G
120 000132 000407
121
122 000134 016211 000000G
123 000140 000401
124 000142 011211
125
126 000144 021200
127 000146 001375
128 000150 012612
129 000152
130
131 000152
132 000152
133
134 000154 016700 000002'
135 000160 001404
136 000162 016701 000000'

;+
$XLBR - RELEASE LIBRARY REFERENCE
:
: INPUTS:
:   R0=LIBRARY DESCRIPTOR BIAS
:   R2=LIBRARY DESCRIPTOR OFFSET INTO BLOCK
:
: OUTPUTS:
:   R0,R1,R2=DESTROYED
:
$XLBR::
BIC #*C<77>,R2 ; USE ONLY LOW 6 BITS OF OFFSET
MOV R2,LBADR ; SAVE OFFSET VALUE
CLR LBALL ; NO LIBRARY TO DE-ALLOCATE
CLR LBIAS ; NO LIBRARY BLOCK TO DE-ALLOCATE
MOV @NLPT,NLBLK ; SAVE NETLDR DATA BLOCK BIAS

SWSTK$ 60$ ;* ENTER KERNEL MODE

CALL $MAPX ;* MAP TO LIBRARY DESCRIPTOR
BIS LBADR,R2 ;* RESTORE LOW 6 BITS OF ADDRESS
DECB LB,USE(R2) ;* DECREMENT USE COUNT
BNE 50$ ;* IF NE, STILL SOME ACTIVE REFERENCES
CLR (R2) ;* ZERO THE FIRST WORD OF DESCRIPTOR
MOV LB,ADR(R2),LBALL ;* SAVE LIBRARY BIAS AND SIZE FOR LATER
MOV LB,SIZE(R2),LBSIZ ;*
BIC #77,R2 ;* RESET TO START OF 32. WORD BLOCK
CMP NLBLK,R0 ;* WAS DESCRIPTOR WITHIN NETLDR DATA BLOCK?

BEQ 50$ ;* IF EQ, YES
DEC 2(R2) ;* ADJUST DESCRIPTOR COUNT FOR THIS BLOCK
BNE 50$ ;* IF NE, BLOCK IS STILL ACTIVE
MOV (R2),-(SP) ;* SAVE POINTER TO NEXT BLOCK

MOV R0,LBIAS ;* SAVE BIAS OF THIS BLOCK
MOV NLBLK,(R1) ;* MAP TO THE DATA BLOCK

CMP .CXLBR(R2),R0 ;* IS OURS THE FIRST LIBRARY BLOCK?
BNE 10$ ;* IF NE, NO
MOV (SP)+,.CXLBR(R2) ;* SET NEW FIRST LIBRARY BLOCK ADDRESS
BR 40$ ;*

10$: MOV .CXLBR(R2),(R1) ;* MAP TO THE FIRST LIBRARY BLOCK
BR 30$ ;*
20$: MOV (R2),(R1) ;* MAP TO THE NEXT LIBRARY BLOCK

30$: CMP (R2),R0 ;* DOES THIS BLOCK POINT AT OURS?
BNE 20$ ;* IF NE, NO
MOV (SP)+,(R2) ;* CHANGE POINTER TO NEXT BLOCK

40$:
50$:
RETURN ;* BACK TO USER MODE

60$: MOV LBALL,R0 ; GET LIBRARY BIAS
BEQ 70$ ; IF ZERO, NO WORK TO DO
MOV LBSIZ,R1 ; GET LIBRARY SIZE

```

.TITLE XXBUF - END OF TASK BUFFER INITIALIZATION
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

END-OF-TASK BUFFER

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RX V1.0

TTTTTTTTTT	MM	MM	PPPPPPPP	EEEEEEEEEE	XX	XX	555555555	
TTTTTTTTTT	MM	MM	PPPPPPPP	EEEEEEEEEE	XX	XX	555555555	
TT	MMMM	MMMM	PP	EE	XX	XX	55	
TT	MMMM	MMMM	PP	EE	XX	XX	55	
TT	MM	MM	PP	EE	XX	XX	555555	
TT	MM	MM	PP	EE	XX	XX	555555	
TT	MM	MM	PPPPPPPP	EEEEEEEEEE			55	
TT	MM	MM	PPPPPPPP	EEEEEEEEEE			55	
TT	MM	MM	PP	EE	XX	XX	55	
TT	MM	MM	PP	EE	XX	XX	55	
TT	MM	MM	PP	EE	XX	XX	55
TT	MM	MM	PP	EE	XX	XX	55
TT	MM	MM	PP	EE	XX	XX	555555
TT	MM	MM	PP	EE	XX	XX	555555

LL	SSSSSSSS	TTTTTTTTTT
LL	SSSSSSSS	TTTTTTTTTT
LL	SS	TT
LL	SS	TT
LL	SS	TT
LL	SS	TT
LL	SSSSSS	TT
LL	SSSSSS	TT
LL	SS	TT
LL	SS	TT
LL	SS	TT
LL	SS	TT
LL	SS	TT
LLLLLLLLLL	SSSSSSSS	TT
LLLLLLLLLL	SSSSSSSS	TT

```
571  
572 001074 005203      :  
573 001074      :DPRB:  INC    R3  
574      :      :      RETURN  
575      :  
576      :DEAD POLLING RATIO (DPRW) - WORD  
577      :  
578 001100 005723      :DPRW:  TST    (R3)+  
579 001102      :      :      RETURN
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49

.TITLE TMPEX6 - NTL TEMPLATE EXPANSION ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

NTL - END OF PROCESS LINE TABLE EXPANSION ROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

```

596
597
598
599 001222 012724 013746
600 001226 016724 000000G
601 001232 012724 012737
602 001236 016724 000202
603 001242 016724 000000G
604 001246 012724 004537
605 001252 112524
606 001254 112524
607
608
609
610
611 001256 016700 000172'
612 001262 016746 000000G
613 001266 066016 000000G
614 001272 062716 000000G
615 001276 042716 160000
616 001302 056716 000154'
617 001306 016703 000000G
618 001312 066703 000170'
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652

; INTERRUPT ENTRY POINT
INT:  MOV    #13746,(R4)+ ; STORE MOV @#KISAR5,-(SP)
      MOV    K SAR5,(R4)+ ;
      MOV    #12737,(R4)+ ; STORE "MOV BIAS,@#KISAR5"
      MOV    PRBIAS,(R4)+ ;
      MOV    K SAR5,(R4)+ ;
      MOV    #4537,(R4)+ ; STORE "JSR R5,@#ENTRY"
      MOVB   (R5)+,(R4)+ ;
      MOVB   (R5)+,(R4)+ ;

; SET UP ICB
      MOV    NXTLN,R0 ; GET COUNT OF CURRENT ".INT" * 2
      MOV    $ADDR,-(SP) ; GET ADDRESS OF LINE TABLE ON STACK
      ADD    $VECT1(R0),(SP) ; ADD IN THE DISPLACEMENT VALUE
      ADD    # $JSR0F,(SP) ; ... AND SKIP TO THE JSR INSTRUCTION
      BIC    #160000,(SP) ; MAKE SURE WE USE KERNEL APR5
      BIS    PRAPR,(SP) ;
      MOV    $ICBAD,R3 ; GET ADDRESS OF ICB
      ADD    ICBPT,R3 ; ADD IN THE DISPLACEMENT FOR CURRENT ICB

      .IF DF R$SMPL
      MOV    (R1),-(SP) ; SAVE CURRENT MAPPING
      MOV    R2,-(SP) ; SAVE CURRENT BASE ADDRESS
      CLR    -(SP) ; SAVE PLACE FOR ICB POINTER
      TSTB   @NCPU ; IS THIS A MULTIPROCESSOR SYSTEM ?
      BEQ    20$ ; IF EQ, NO

      MOV    PRSLTA,R2 ; GET THE SLT ADDRESS
      MOV    L,KRBA(R2),R2 ; TO GET THE KRB ADDRESS
      ADD    #K,PRM,R2 ; GET START OF ICB CHAIN
10$:  MOV    (R2),R2 ; GET NEXT ICB
      BEQ    30$ ; IF EQ, NO MORE

      MOV    R2,R3 ; COPY ICB LINK POINTER
      BIC    #7777,R2 ; CLEAR OUT ALL BUT CPU NUMBER
      BIC    R2,R3 ; CLEAN OUT CPU NUMBER
      SWAB   R2 ; FORM CPU NUMBER*2
      ASR    R2 ;
      ASR    R2 ;
      ASR    R2 ;
      MOV    K6TAB,TEMP ; COPY $K6TAB ADDRESS
      ADD    R2,TEMP ; GET CORRECT CPU BIAS ADDRESS
      MOV    @TEMP,(R1) ; COPY CPU BIAS
      ASL    R3 ; CONVERT ICB POINTER TO VIRTUAL ADDRESS
      BIS    #120000,R3 ; FORCE APR5 MAPPING
      ; NOTE - FOR 11M+, ASSUME APR5 IS LOW APR
      MOV    R3,(SP) ; SAVE ICB POINTER
      ADD    ICBPT,R3 ; ADD IN THE DISPLACEMENT TO CURRENT ICB
20$:  TST    (R3)+ ; SKIP THE LINK POINTER

      .IFTF

```


TMPEX6 CREATED BY MACRO ON 29-JUN-85 AT 00:43 PAGE 2 J 5

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
LN.LOA	= 000004	#5-58
LN.LOO	= 000003	#5-58
LN.OAU	= 000003	#5-58
LN.OFF	= 000001	#5-58
LN.ON	= 000000	#5-58
LN.OOP	= 000004	#5-58
LN.OPE	= 000001	#5-58
LN.REF	= 000002	#5-58
LN.SER	= 000002	#5-58
LN.STA	= 000017	#5-58
LN.SUB	= 000360	#5-58
LN.TRI	= 000006	#5-58
LSTHD	000462 R	6-109 #8-410
LTAB	000500 R	6-110 #8-420
L.COST	000015	#5-58
L.CTL	000012	#5-58
L.CVA	177776	#5-58 *8-371
L.DDM	000002	#5-58
L.DDS	000004	#5-58
L.DLC	000003	#5-58
L.DLM	000006	#5-58
L.DLS	000010	#5-58
L.FLG	000000	#5-58
L.KRBA	000016	#5-58
L.LEN	= 000022	#5-58
L.MPF	000022	#5-58
L.NMST	000020	#5-58
L.NSTA	000014	#5-58
L.OWNR	000021	#5-58
L.UNT	000013	#5-58
MPTAB	001474 R	6-111 #12-738
MXPTB	001754 R	6-120 #14-812
MXTAB	001406 R	6-112 #12-716
NOD	000554 R	6-113 #8-443
NXTLN	000172 R	#6-183 *7-261 11-611 *11-701
PAPR	000432 R	#6-224 14-862
PECHA	000404 R	6-130 #8-383
PRAPR	000154 R	#6-152 7-281 8-412 11-616 12-757 12-761 12-765 14-814 *14-816
		*14-837
PRBIAS	000202 R	#6-191 11-602 11-668
PRCSR	000343 R	#6-219 8-346
PRCTIM	000225 R	#6-199 8-352
PRDCHA	000556 R	#6-228 8-373 8-374
PRDDPV	000256 R	#6-216
PRDEV	000214 R	#6-196
PRDLPV	000254 R	#6-215
PRDPR	0005 R	#6-225 14-871 14-877
PRFLAG	000142 R	#6-175 *7-271 8-300 8-310 8-359 8-361 8-384 8-386 8-429
PRI	001374 R	6-114 #11-709
PRINC	000206 R	#6-194
PRLINX	000220 R	#6-198 8-404
PRLSAD	001224 R	#6-232 8-302 8-313

```

337
338
339
340 000442 005205
341 000444 005723
342 000446
343
344
345
346 000450 005723
347 000452
348
349
350
351
352 000454 005767 000352'
353 000460 001417
354 000462 126467 000013 000350'
355 000470 001013
356 000472
357 000474 016702 000000G
358 000500 010362 177776
359 000504
360 000506 016723 000214'
361 000512 016723 000216'
362 000516 000401
363 000520 022323
364 000522
365
366
367
368
369 000524 005767 000352'
370 000530 001407
371 000532 126467 000013 000350'
372 000540 001003
373 000542 016723 000224'
374 000546 000401
375 000550 005723
376 000552
377
378
379
380
381 000554
382
383
384
385
386 000556 122525
387 000560 062703 000016
388 000564
389
390
391
392
393 000566

; CSR
CSR: INC R5
      TST (R3)+
      RETURN

; COUNTER TIMER
CTIM: TST (R3)+
      RETURN

; DEVICE CHARACTERISTICS
DVCHA: TST FLAGS ; IS IT MUX EXPANSION ?
        BEQ 10$ ; NO
        CMPB L.UNT(R4),INDEX ; DO UNIT NUMBERS MATCH ?
        BNE 10$ ; NO
        SAVRG R2 ; SAVE REG
        MOV $SLTSV,R2 ; COPY SL1 ADDRESS
        MOV R3,L.CVA(R2) ; STORE VIRTUAL ADDRESS POINTER
        RESRG R2 ; RESTORE REG
        MOV PRDCHA,(R3)+ ; COPY DEVICE CHARACTERISTICS
        MOV PRDCHA+2,(R3)+ ; TWO WORDS
        BR 20$
10$: CMP (R3)+,(R3)+
20$: RETURN

; PROTOCOL EMULATOR CHARACTERISTICS
PECHA: TST FLAGS ; IS IT MUX EXPANSION ?
        BEQ 10$ ; NO
        CMPB L.UNT(R4),INDEX ; DO UNIT NUMBERS MATCH ?
        BNE 10$ ; NO
        MOV PRPCHA,(R3)+ ; COPY PROTOCOL EMULATOR CHARACTERISTICS
        BR 20$
10$: TST (R3)+
20$: RETURN

; END OF FILE
EOF: CRASH

; INTERRUPT ENTRY POINT
INT: CMPB (R5)+,(R5)+
      ADD #14,R3
      RETURN

; LIBRARY POINTER
LIBR:

```

```

Symbol table

ADDRB 000360R    K$LDLC= 000000    L$11R= 000000    PRSLTA 000220R    002 UBIAS 000354R    002
ADDRW 000364R    K$TPS= 000074    L.COST 000015    PRTOV 000606R    UNB 000760R
APR 001346R      K.CSR 000002    L.CTL 000012    PR2HLD 000326R    002 UNW 000764R
A$CHK= 000000    K.PRI 000000    L.CVA 177776    PR2MBS 000246R    002 V$SCTR= 001000
A$CPS= 000000    K.VCT 000001    L.DDM 000002    PR2MRT 000266R    002 WORD 000770R
A$PRI= 000000    LD$LP = 000000    L.DDS 000004    PR2MWS 000226R    002 X$SDBT= 000000
A$TRP= 000000    LFILE 001112R    L.DLC 000003    PR2RET 000306R    002 X2CH01 001644R
BLKB 000370R      LF.ACT= 100000    L.DLM 000006    PR7 = ***** GX    X2CH02 001700R
BLKW 000406R      LF.BRO= 000400    L.DLS 000010    P$P45= 000000    X2CH03 001734R
BLOCK 000360R    002 LF.BWT= 000007    L.FLG 000000    P$WRD= 000000    X2CH04 001770R
BYTE 000426R      LF.ENA= 002000    L.KRBA 000016    Q$OPT= 000010    X2CH05 002024R
CORE 000432R      LF.LPB= 001000    L.LEN = 000022    R$SDE= 000000    X3CH01 001570R
CSR 000442R       LF.MDC= 000100    L.MPF 000022    R$SK11= 000001    X3CH02 001574R
CTIM 000450R      LF.MFL= 004000    L.NMST 000020    R$SND= 000000    X3CH03 001600R
C$CKP= 000000    LF.MTP= 000020    L.NSTA 000014    R$11M= 000000    X3CH04 001604R
C$DRE= 000400    LF.PAC= 000200    L.OWNR 000021    SCOM 000656R    X3CH05 001610R
C$RSH= 177564    LF.RDY= 040000    L.UNT 000013    SECSR 000666R    X3CH06 001614R
DPRB 001352R      LF.REA= 010000    MPTAB 001110R    SF.ACT= 000200    X3CH07 001620R
DPRW 001356R      LF.SER= 000040    MXPBT 001174R    SF.ENA= 000100    X3CH08 001624R
DVCHA 000454R      LF.TIM= 000010    MXTAB 000776R    SF.LPB= 000004    X3CH09 001630R
D$BUG= 177514    LF.UNL= 020000    M$CRB= 000124    SF.MFL= 000040    X3CH10 001634R
D$TSK= 000000    LF.X2P= 000000    M$CRX= 000000    SF.PAC= 000020    X3CH11 001640R
D$11= 000001    LIBR 000566R    M$FCS= 000000    SF.REA= 000010    ZTIME = ***** GX
D$YNC= 000000    LINKS 000576R    M$MGE= 000000    SF.SER= 000001    $ADDR = ***** GX
D$YNM= 000000    LN.CLO= 000000    M$NET= 000000    SF.SVC= 000002    $BIAS = ***** GX
EOF 000554R       LN.DUM= 000005    M$OVR= 000000    SF.UNL= 000040    $BIN = ***** GX
E$XPR= 000000    LN.LOA= 000004    MOD 000602R    $FLAGS = ***** GX
FILE 001002R      LN.LOD= 000003    NTLPT = ***** GX    $MAPX = ***** GX
FLAGS 000352R    002 LN.OAU= 000003    N$ACC= 000001    SLNB 000740R    $PTR = ***** GX
FLT 001362R       LN.OFF= 000001    N$BUF= 000001    SLNW 000750R    $SLTA = ***** GX
FLT16 001362R     LN.ON = 000000    N$LDV= 000001    STNB 000754R    $SLTSV= ***** GX
FL.KMX= ***** GX    LN.OOP= 000004    N$MCP= 000001    STNW 000754R    $TMEX7 000000RG
F$LVL= 000001    LN.OPE= 000001    N$MLL= 000001    S$WRG= 000000    $X2HLD= ***** GX
G$TPP= 000000    LN.REF= 000002    N$MOV= 000010    S$YSZ= 007600    $X2MBS= ***** GX
G$TSS= 000000    LN.SER= 000002    N$NCT= 000001    S.COST 000001    $X2MRT= ***** GX
G$TTK= 000000    LN.STA= 000017    N$PEM= 000001    S.FLG 000000    $X2MWS= ***** GX
G$WRD= 000000    LN.SUB= 000360    PECHA 000524R    S.LEN 000004    $X2RET= ***** GX
INDEX 000350R    002 LN.TRI= 000006    PRDCHA 000214R    S.NMST 000002    $DCHA= ***** GX
INT 000556R       LSTHD 000614R    PRI 000624R    S.OWNR 000003    $PCHA= ***** GX
I$RAR= 000000    LTAB 000620R    PRFLAG 000346R    TABLE 000002R    $PRI = ***** GX
I$RDN= 000000    L$ASG= 000000    PRPCHA 000224R    TEMP 000000R    $SCSR= ***** GX
K$CNT= 177546    L$DRV= 000000    PRPRI 000222R    TIME 001314R    $CXUNL= ***** GX
K$CSR= 177546    L$P11= 000001    PRSCSR 000154R    T$KMG= 000000    .EOF = ***** GX
    T$MIN= 000000    UBA 000356R

```

```

. ABS. 177776 000 (RW,I,GBL,ABS,OVR)
        002060 001 (RW,I,LCL,REL,CON)
DATA 000460 002 (RW,D,LCL,REL,CON)
Errors detected: 0

```

*** Assembler statistics

```

Work file reads: 0
Work file writes: 0
Size of work file: 10943 Words ( 43 Pages)
Size of core pool: 14440 Words ( 55 Pages)

```

245
246
247

000001

.END

163 000002	TRANS	"STNW",NONE,\$STNW	
164 000002	TRANS	"UNW",NONE,\$UNW	
165 000002	TRANS	"WORD",WORD	
166 000002	TRANS	"DPRW",NONE,\$DPRW	
167 000002	TRANS	"TIME",NONE,\$TIME	
168 000002	TRANS	"MXPTB",MXPTB	
169 000002	TRANS	"X3CHW",X3CHW	
170 000002	TRANS	"X2CHW",X2CHW	
171 000002	TRANS	\$LAMDA,\$EXIT,ERR\$1	; "UNDEFINED OPERATOR"
172			

TMPSTB CREATED BY MACRO ON 29-JUN-85 AT 00:46 PAGE 2 J 10
 MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

DBGTP\$	#7-75	#7-82	#7-83	#7-96	#7-103	#8-113	#8-114	#8-115	#8-116	#8-117
	#8-118	#8-119	#8-120	#8-121	#8-122	#8-123	#8-124	#8-125	#8-126	#8-127
	#8-128	#8-134	#8-135	#8-136	#8-137	#8-138	#8-139	#8-140	#8-141	#8-142
	#8-143	#8-144	#8-145	#8-146	#8-147	#8-148	#8-149	#8-150	#8-151	#8-152
	#8-153	#8-154	#8-155	#8-156	#8-157	#8-158	#8-159	#8-160	#8-161	#8-162
	#8-163	#8-164	#8-165	#8-166	#8-167	#8-168	#8-169	#8-170	#8-171	#9-181
	#9-184	#9-191	#9-202	#9-209	#10-218	#10-219	#10-222	#10-225	#10-226	#10-233
	#10-236	#10-243	#11-262	#11-263	#11-266	#11-269	#11-276	#11-279	#11-282	#12-292
	#12-299	#12-300	#12-303	#12-310	#12-321	#12-326	#13-336	#13-345	#13-355	#14-365
	#14-372	#14-373	#14-380	#14-388	#14-393	#14-400	#15-408	#15-409	#15-412	#15-415
	#15-416	#16-444	#16-447	#16-462	#16-463	#16-480	#16-487	#16-494	#16-501	#16-517
ISTAT\$	#5-60									
MTRANS\$	#7-75	7-75								
RETURN	6-68									
STAT\$	#5-60	7-80	#7-88	#7-94	#7-101	8-111	#8-132	9-179	#9-182	#9-185
	9-193	9-200	#9-207	10-216	#10-220	#10-223	#10-231	#10-234	#10-237	10-245
	11-254	#11-260	#11-264	#11-267	#11-274	#11-277	#11-280	12-290	#12-297	#12-301
	#12-304	12-312	12-319	#12-322	#12-324	#12-327	13-334	#13-337	#13-343	#13-346
	#13-353	#13-356	14-363	#14-370	#14-378	#14-385	#14-398	15-406	#15-410	#15-413
	#15-420	#15-425	#15-430	#15-435	16-442	#16-445	#16-448	#16-454	#16-460	#16-464
	#16-470	#16-472	#16-478	#16-485	#16-492	#16-499	16-515	#16-522		
TRANS\$	#5-60	#7-81	#7-82	#7-83	#7-89	#7-95	#7-96	#7-102	#7-103	#8-112
	#8-113	#8-114	#8-115	#8-116	#8-117	#8-118	#8-119	#8-120	#8-121	#8-122
	#8-123	#8-124	#8-125	#8-126	#8-127	#8-128	#8-133	#8-134	#8-135	#8-136
	#8-137	#8-138	#8-139	#8-140	#8-141	#8-142	#8-143	#8-144	#8-145	#8-146
	#8-147	#8-148	#8-149	#8-150	#8-151	#8-152	#8-153	#8-154	#8-155	#8-156
	#8-157	#8-158	#8-159	#8-160	#8-161	#8-162	#8-163	#8-164	#8-165	#8-166
	#8-167	#8-168	#8-169	#8-170	#8-171	#9-180	#9-181	#9-183	#9-184	9-187
	9-191	#9-194	#9-201	#9-202	#9-208	#9-209	#10-217	#10-218	#10-219	#10-221
	#10-222	#10-224	#10-225	#10-226	#10-232	#10-233	#10-235	#10-236	10-239	10-243
	#10-246	#11-255	#11-261	#11-262	#11-263	#11-265	#11-266	#11-268	#11-269	#11-275
	#11-276	#11-278	#11-279	#11-281	#11-282	#12-291	#12-292	#12-298	#12-299	#12-300
	#12-302	#12-303	12-306	12-310	#12-313	#12-320	#12-321	#12-323	#12-325	#12-326
	#12-328	#13-335	#13-336	#13-338	#13-344	#13-345	#13-347	#13-354	#13-355	#13-357
	#14-364	#14-365	#14-371	#14-372	#14-373	#14-379	#14-380	14-387	#14-388	14-393
	#14-399	#14-400	#15-407	#15-408	#15-409	#15-411	#15-412	#15-414	#15-415	#15-416
	#15-421	#15-426	#15-431	#15-436	#16-443	#16-444	#16-446	#16-447	#16-449	#16-455
	#16-461	#16-462	#16-463	#16-465	#16-471	#16-473	#16-479	#16-480	#16-486	#16-487
	#16-493	#16-494	#16-500	#16-501	#16-516	#16-517				

```

AB = ***** GX
A$$$CHK= 000000
A$$$CPS= 000000
A$$$PRI= 000000
A$$$TRP= 000000
C$$$CKP= 000000
C$$$ORE= 000400
C$$$RSH= 177564
D$$$BUG= 177514
D$$$ISK= 000000
D$$$L11= 000001
D$$$YNC= 000000
D$$$YNM= 000000
E$$$XPR= 000000
FL.CHA= ***** GX
FL.DDM= ***** GX
FL.DLC= ***** GX
FL.MUX= ***** GX
FMKTMP 000024R
FMT1 000062R
FM.1 = 000000
F$$$LVL= 000001
F1.VSE= ***** GX
G$$$TPP= 000000
G$$$TSS= 000000
G$$$TTK= 000000
G$$$WRD= 000000
I$$$RAR= 000000
I$$$RDN= 000000
K$$$CNT= 177546
K$$$CSR= 177546
K$$$LDC= 000000
K$$$TPS= 000074
K6TMP 000020R

LD$LP= 000000
LF.ACT= 100000
LF.BRO= 000400
LF.BWT= 000007
LF.ENA= 002000
LF.LPB= 001000
LF.MDC= 000100
LF.MFI= 004000
LF.MTP= 000020
LF.PAC= 000200
LF.RDY= 040000
LF.REA= 010000
LF.SER= 000040
LF.TIM= 000010
LF.UNL= 020000
LF.X2P= 000000
LN.CLD= 000000
LN.DUM= 000005
LN.LDA= 000004
LN.LOG= 000003
LN.OAU= 000003
LN.OFF= 000001
LN.ON= 000000
LN.ODP= 000004
LN.OPE= 000001
LN.REF= 000002
LN.SER= 000002
LN.STA= 000017
LN.SUB= 000360
LN.TR1= 000006
L$$$ASG= 000000
L$$$DRV= 000000

L$$$P11= 000001
L$$$11R= 000000
L.COST 000015
L.CTL 000012
L.CVA 177776
L.DDM 000002
L.DDS 000004
L.DLC 000003
L.DLM 000006
L.DLS 000010
L.FLG 000000
L.KRBA 000016
L.LEN= 000022
L.MPF 000022
L.NMST 000020
L.NSTA 000014
L.OWNR 000021
L.UNT 000013
MANINI 000642R
M$$$CRB= 000124
M$$$CRX= 000000
M$$$FCS= 000000
M$$$MGE= 000000
M$$$NET= 000000
M$$$OVR= 000000
NSITAB 000000R
NSITMP 000022R
NS0 = ***** GX
NS1 = ***** GX
NS2 = ***** GX
NS3 = ***** GX
NS4 = ***** GX
NS5 = ***** GX

NS6 = ***** GX
NS7 = ***** GX
N$$$ACC= 000001
N$$$BUF= 000001
N$$$LDV= 000001
N$$$MCP= 000001
N$$$MLL= 000001
N$$$MOV= 000010
N$$$NCT= 000001
N$$$PEM= 000001
PR7 = ***** GX
PS = ***** GX
P$$$P45= 000000
P$$$WRD= 000000
Q$$$OPT= 000010
R$$$DER= 000000
R$$$K11= 000001
R$$$SND= 000000
R$$$11M= 000000
SF.ACT= 000200
SF.ENA= 000100
SF.LPB= 000004
SF.MFL= 000040
SF.PAC= 000020
SF.REA= 000010
SF.SER= 000001
SF.SVC= 000002
SF.UNL= 000040
SLTMA = ***** GX
SLTMM = ***** GX
S$$$WRG= 000000
S$$$YSZ= 007600
S.COST 000001

S.FLG 000000
S.LEN 000004
S.NMST 000002
S.OWNR 000003
T$$$KMG= 000000
T$$$MIN= 000000
V$$$CTR= 001000
X$$$DBT= 000000
ZF.PSE= ***** GX
Z.DAT = ***** GX
Z.DSP = ***** GX
Z.FLG = ***** GX
Z.PCB = ***** GX
$ADDR = ***** GX
$DDMPV= ***** GX
$DLCPV= ***** GX
$ERR38 000026R
$FLAGS= ***** GX
$FLAG1= ***** GX
$ICBAD= ***** GX
$MXADD= ***** GX
$MXLEN= ***** GX
$NVECT= ***** GX
$OFF = ***** GX
$PDVA = ***** GX
$REP.P= ***** GX
$SLTA = ***** GX
$TERR = ***** GX
$TMLTA 000250RG
$TMRDY 000564RG
$TMVCT 000000RG
$VECT1= ***** GX
$VVECT= ***** GX

```

002

```

. ABS. 177776 000 (RW,I,GBL,ABS,OVR)
000772 001 (RW,I,LCL,REL,CON)
DATA 000066 002 (RW,D,LCL,REL,CON)
Errors detected: 0

```

*** Assembler statistics

```

Work file reads: 0
Work file writes: 0
Size of work file: 12005 Words ( 47 Pages)
Size of core pool: 14440 Words ( 55 Pages)
Operating system: RSX-11M/PLUS

```

```

Elapsed time: 00:00:13.25
SY:TMPVCT.V2,[132,134]TMPVCT/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]TMPVCT

```

UNLHC CREATED BY MACRO ON 29-JUN-85 AT 00:48 PAGE 2 J 12

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL VALUE REFERENCES

T\$NFE	000000	#5-52				
T\$NLEN	000010	#5-52				
T\$NNUL	000002	#5-52				
T\$NCP	000006	#5-52				
T\$NRNI	000042	#5-52				
T\$NRPL	000005	#5-52				
T\$NRUL	000007	#5-52				
T\$NVR	000001	#5-52				
T\$RPR	000040	#5-52				
T\$SVC	000034	#5-52				
T\$T5	000030	#5-52				
T\$T6	000032	#5-52				
\$DEAHC	000000 RG	#6-71				
\$DEA18	= ***** GX	6-84	6-89	6-99	6-109	6-117
\$MUL	= ***** GX	6-78	6-103			


```

320                                     .SBTTL XTRA - DEALLOCATE TRACE DATA STRUCTURES
321
322                                     ;*
323                                     XTRA - DEALLOCATE TRACE STRUCTURES
324                                     :
325                                     INPUTS
326                                     NONE
327                                     :
328                                     OUTPUTS
329                                     R0, R1, R2 DESTROYED
330                                     TRACE STRUCTURES ARE DEALLOCATED
331                                     :
332                                     -
333                                     .IF NDF I$$AS
334
335                                     XTRA:
336                                     SWSTK$ 30$                ;; ENTER SYSTEM STATE
337
338                                     MOV    @PSIPT,R3          ;; GET HOME BLOCK ADDRESS
339                                     MOV    HSTRB(R3),R0        ;; GET TRACE CONTROL BLOCK ADDRESS
340                                     BEQ    30$                ;; BR IF NOT TRACING
341                                     CLR    HSTRB(R3)          ;; CAUSE TRACE TASK TO EXIT
342                                     MOV    T$TCB(R0),R1        ;; GET TRACE TASK TCB
343                                     BEQ    10$                ;; IF EQ, NONE
344                                     BIC    #T2.STP*2!T2.STP,T.ST2(R1) ;; UNSTOP TRACE TASK
345
346                                     10$: SAVRG <R0>            ;; SAVE ADDRESS OF BLOCK
347                                     SAVRG <#T$CLEN>          ;; SAVE LENGHYH OF SAME
348
349                                     MOV    T$CSIZ(R0),R1      ;; GET SIZE OF BUFFER IN BLOCKS
350                                     ASL$   6,R1              ;; COMPUTE SIZE IN BYTES
351                                     MOV    R0,R2              ;; COPY END POINTER
352                                     ADD    (SP),R2            ;; COMPUTE END ADDRESS OF TRACE BLOCK
353                                     CMP    T$BUF(R0),R2        ;; IS BUFFER CONTIGUOUS WITH CONTROL BLOCK?
354                                     BEQ    20$                ;; YES, DEALLOCATE AS ONE
355
356                                     ;*** DEALLOCATE NON-POOL TRACE BUFFER HERE (HOW?) ***
357                                     BR     25$                ;; MERGE TO DEALLOCATE CONTROL BLOCK
358
359                                     20$: ADD    R1,(SP)        ;; INCLUDE LENGTH OF BUFFER IN DEALLOC
360                                     25$: RESRG <R1>          ;; GET LENGTH TO DEALLOCATE
361                                     RESRG <R0>              ;; GET CONTROL BLOCK ADDRESS
362                                     CALL   $DECEX            ;; DEALLOCATE TO POOL
363                                     30$: RETURN              ;; RETURN TO USER MODE AND CALLER
364
365                                     .ENDC
366
367                                     0G0001 .END
  
```

UNLSUB - NTL UNLOAD SUBROUTINES MACRO V05.03b Saturday 29-Jun-85 00:48 Page 9

GETPDV - CONVERT PDV INDEX TO PDV ADDRESS

163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179

000044 067704 000000G
000050 016404 000000G
000054

```
.SBTTL  GETPDV - CONVERT PDV INDEX TO PDV ADDRESS
;+
; GETPDV - CONVERT PDV INDEX TO PDV ADDRESS
; INPUTS:
;   R4 - PDV INDEX
; OUTPUTS:
;   R4 - PDV ADDRESS
;-
GETPDV::ADD  @PDVTA,R4      ; ADD PDV TABLE ADDRESS
            MOV  .CBIAS(R4),R4 ; GET TABLE ENTRY
10$:      RETURN
```

K 15

```

56 000000          .PSECT DATA,D
57
58          001130          $BFLEN==600.
59          :
60          : POOL LIMITS
61
62 000000 000004'          $XXBUF::WORD XXBUF          ; STARTING ADDRESS
63 000002 001130          $XXLEN::WORD $BFLEN          ; LENGTH IN BYTES
64
65 000004          XXBUF: .BLKB $BFLEN
66          :
67          : "CORAL" POINTERS
68          :
69 001134 000003          .WORD 3          ; ROUNDING FACTOR
70 001136 000004'          $XXAVL::WORD XXBUF          ; POINTER TO FIRST FREE BLOCK
71 001140 000000          .WORD 0

```

TMPEX5 - NTL LINE TABLE UPDATE MACRO V05.03b Saturday 29-Jun-85 ^{K 1} 00:43
Table of contents

5-	55	MACRO DEFINITIONS
11-	656	X3CH
12-	714	X2CH

```

581
582
583
584
585
586
587
588
589
590
591
592
593
594 001104
595 001104 005067 000160'
596 001110 017700 000000G
597 001114
598
599 001120
600 001124 016211 000000G
601 001130 001470
602
603 001132 005722
604 001134 005722
605 001136 126204 177776
606 001142 001013
607 001144 022203
608 001146 001443
609 001150 101011
610
611 001152 011246
612 001154 042716 000003
613 001160 066216 177776
614 001164 022603
615 001166 101033
616 001170 000401
617
618 001172 005722
619 001174 032722 000001
620 001200 001470
621 001202 005046
622 001204 152216
623 001206 005046
624 001210 152216
625
626 001212 062616
627 001214 006316
628 001216 006316
629 001220 062602
630 001222 032702 000077
631 001226 001403
632 001230 005712
633 001232 001340
634 001234 000402
635
636 001236 162702 000100
637 001242 042702 000077

+
FLT16 - FIND A 16 BIT ALLOCATION LINE TABLE
:
: INPUTS:
: R3=LINE TABLE ADDRESS
: R4=PDV INDEX OF DDM/DLC/LLC PROCESS
:
: OUTPUTS:
: R0,R1,R2,R3=DESTROYED
: UBIAS,UBA=UNLOAD BLOCK ADDRESS
: BLOCK=LOCAL COPY OF UNLOAD BLOCK
:
-

FLT16:
FLT: CLR UBA ; INDICATE NO UNLOAD BLOCK FOUND YET
MOV @NTLPT,R0 ; GET NETLDR DATA BLOCK BIAS
SWSTK$ 100$ ; * ENTER KERNEL MODE

CALL $MAPX ; * MAP TO THE DATA BLOCK
MOV ,CXUNL(R2),(R1) ; * AND TO THE FIRST UNLOAD BLOCK
BEQ 100$ ; * IF ZERO, NO UNLOAD BLOCKS

10$: TST (R2)+ ; * SKIP POINTER TO NEXT UNLOAD BLOCK
20$: TST (R2)+ ; * POINT PAST PDV INDEX
CMPB -2(R2),R4 ; * IS THIS THE RIGHT PDV?
30$ BNE 30$ ; * IF NE, NO
CMP (R2)+,R3 ; * IS THIS OUR LINE TABLE?
BEQ 80$ ; * IF EQ, YES
BHI 40$ ; * IF HI, NO

MOV (R2),-(SP) ; * EXTRACT SIZE FIELD
BIC #3,(SP) ; *
ADD -2(R2),(SP) ; * ADD TABLE ENTRY
CMP (SP)+,R3 ; * IS THIS OUR LINE TABLE?
BHI 80$ ; * IF HI, YES
BR 40$ ; * ELSE, SKIP ANY SUB-ALLOCATIONS

J$: TST (R2)+ ; * SKIP THE LINE TABLE ADDRESS
40$: BIT #1,(R2)+ ; * ANY SUB-ALLOCATIONS THIS ENTRY?
BEQ 50$ ; * IF EQ, NO
CLR -(SP) ; * GET NUMBER OF ALLOCATIONS
BISB (R2)+,(SP) ; *
CLR -(SP) ; * AND NUMBER OF LIBRARIES
BISB (R2)+,(SP) ; *
...

ADD (SP)+,(SP) ; * ADD THEM TOGETHER
ASL (SP) ; * MULTIPLY BY 4
ASL (SP) ; *
ADD (SP)+,R2 ; * GIVING NEXT TABLE ENTRY
BIT #77,R2 ; * IS IT PHYSICAL END OF BLOCK?
BEQ 60$ ; * IF EQ, YES
TST (R2) ; * IS IT LOGICAL END OF BLOCK?
BNE 20$ ; * IF NE, NO - CHECK NEXT ENTRY
BR 70$ ; *

60$: SUB #100,R2 ; * RESET R2 TO START OF BLOCK
70$: BIC #77,R2 ; *

```

51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78

```
.SBTTL  MACRO DEFINITIONS

:****
: LIBRARY MACROS
:****
      .MCALL  ASL$,SLTDF$,DHBDF$,SAVRG,RESRG
      SLTDF$      ; DEFINE SLT OFFSETS AND SYMBOLS
      DHBDF$      ; DEFINE DECNET HOME BLOCK OFFSETS

      .IF DF  R$$MPL

      .MCALL  KRBDF$

      KRBDF$      ; DEFINE KRB OFFSETS

      .ENDC

:***
: LOCAL MACROS
:***

:
: DEFINE TRANSFER TABLE ENTRY
:
      .MACRO  TTAB$  AA,BB
      .WORD   AA
      .ENDM  TTAB$
```

```

LOCAL DATA
653 001316 012723 013746      MOV    #13746,(R3)+    ; * STORE 'MOV @#KISAR5,-(SP)''
654 001322 016723 000000G     MOV    KSAR5,(R3)+    ; * ...
655
656      .IFT
657
658      MOV    FMASK,TEMP    ; GET $FMASK ADDRESS
659      ADD    #2,TEMP      ; GET $FMASK+2 ADDRESS
660      BIT    #F2.DAS,@TEMP ; IS DATA SPACE ENABLED ?
661      BEQ    25$          ; IF EQ, NO
662      MOV    #13746,(R3)+  ; STORE 'MOV @#KINAR5,-(SP)''
663      MOV    #KINAR5,(R3)+ ; ...
664
25$:
665      .IFTF
666
667 001326 012723 012737      MOV    #12737,(R3)+    ; STORE 'MOV BIAS,@#KISAR5''
668 001332 016723 000202'     MOV    PRBIAS,(R3)+    ; ...
669 001336 016723 000000G     MOV    KSAR5,(R3)+    ; ...
670
671      .IFT
672
673      BIT    #F2.DAS,@TEMP ; IS DATA SPACE ENABLED ?
674      BEQ    27$          ; IF EQ, NO
675      MOV    #12737,(R3)+  ; STORE 'MOV BIAS,@#KINAR5''
676      MOV    PRBIAS,(R3)+  ; ...
677      MOV    #KINAR5,(R3)+ ; ...
678
27$:
679      .IFTF
680
681 001342 012723 000137      MOV    #137,(R3)+    ; STORE 'JMP @#ENTRY IN LINE TABLE''
682
683      .IFT
684
685      MOV    6(SP),(R3)+    ;
686      MOV    (SP),R2        ; GET LINK POINTER
687      BNE    10$           ; IF NE, LOOK FOR NEXT ICB
688      TST    (SP)+          ; CLEAN OFF ICB ADDRESS
689      MOV    (SP)+,R2       ; RESTORE BASE ADDRESS
690      MOV    (SP)+,(R1)     ; RESTORE MAPPING
691      TST    (SP)+          ; CLEAN OFF SERVICE ROUTINE ADDRESS
692
30$:
693      .IFF
694
695 001346 012623      MOV    (SP)+,(R3)+    ; ...
696
697      .ENDC
698
699      ; CLEAN UP FOR NEXT TIME
700
701 001350 062767 000002 000172' ADD    #2,NXTLN      ; SET FOR NEXT LINE
702 001356 016760 000170' 000000G MOV    ICBPT,$VECT1(R0) ; SET UP FOR 'TMPVCT'
703 001364 066767 000174' 000170' ADD    ICBLN,ICBPT    ; ADD IN ANOTHER DISPLACEMENT
704 001372
705      RETURN
706
707      ; COMPLEMENT OF PRIORITY
708
709 001374 012714 000000G     PRI: MOV    #PR7,(R4)    ; STORE PRIORITY 7

```


TMPEX6 CREATED BY MACRO ON 29-JUN-85 AT 00:43 PAGE 3 K 5
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
PRMTP	000350 R	#6-222 8-482 8-495
PRMUX	000514 R	#6-226
PRNAME	000200 R	#6-190
PRPAR	000210 R	#6-195
PRPCB	000204 R	#6-193
PRPCHA	001062 R	#6-230 8-393
PRPDVA	000176 R	#6-189 7-267
PRPRI	000344 R	#6-220 11-710
PRSCSR	000516 R	#6-227 8-435
PRSLN	000260 R	#6-217 8-461 8-473
PRSLTA	000252 R	#6-214
PRSNUM	000352 R	#6-223 8-485 8-499
PRTBL	000216 R	#6-197
PRTOV	000566 R	6-115 #8-450
PRVECT	000346 R	#6-221
PR2HLD	001326 R	#6-237 16-987
PR2MBS	001246 R	#6-234 16-957
PR2MRT	001266 R	#6-235 16-967
PR2MWS	001226 R	#6-233 16-947
PR2RET	001306 R	#6-236 16-977
PR3CAL	000234 R	#6-204 15-905
PR3CLR	000236 R	#6-205 15-910
PR3DBS	000224 R	#6-200 15-885
PR3DWS	000230 R	#6-202 15-895
PR3MBS	000226 R	#6-201 15-890
PR3MCL	000244 R	#6-208 15-925
PR3MRE	000246 R	#6-209 15-930
PR3MRS	000250 R	#6-210 15-935
PR3MWS	000232 R	#6-203 15-900
PR3RES	000240 R	#6-206 15-915
PR3RST	000242 R	#6-207 15-920
PR7	= ***** GX	11-709
R\$SEIS	= *****	9-560
R\$SMPL	= *****	5-61 11-620
R\$S11D	= *****	9-560
R\$S11M	= 000000	6-148
SAVALC	001170 R	#10-584 14-842
SCOM	001132 R	6-116 #10-572
SECSR	000506 R	6-117 #8-426
SF.ACT	= 000200	#5-58
SF.ENA	= 000100	#5-58
SF.LPB	= 000004	#5-58
SF.MFL	= 000040	#5-58
SF.PAC	= 000320	#5-58
SF.REA	= 000010	#5-58
SF.SER	= 000001	#5-58
SF.SVC	= 000002	#5-58
SF.UNL	= 000040	#5-58
SLNB	000572 R	6-118 #8-456
SLNW	000616 R	6-119 #8-467
STLIN	000164 R	#6-177 *7-279 8-420
STNE	000646 R	6-120 #8-480

```

MACRO DEFINITIONS

394 000566 062705 000006      ADD      #6,R5
395 000572 022323              CMP      (R3)+,(R3)+
396 000574                      RETURN
397
398
399      ; LOGICAL LINKS
400
401 000576 005723      LINKS:  TST      (R3)+
402 000600              RETURN
403
404      ; LOCAL NODE ADDRESS
405
406
407 000602 005723      NOD:    TST      (R3)+      ; ADJUST TEMPLATE ADDRESS
408 000604              RETURN
409
410      ; MAXIMUM PROTOCOL SIZE
411
412
413 000606 105723      PRTOV:  TSTB   (R3)+      ; ADJUST TEMPLATE ADDRESS
414 000610 105725              TSTB   (R5)+      ; ADJUST BINARY BUFFER ADDRESS
415 000612              RETURN
416
417      ; LISTHEAD
418
419
420 000614 022323      LSTHD:  CMP      (R3)+,(R3)+
421 000616              RETURN
422
423      ; LINE TABLE ADDRESS
424
425
426 000620 005723      LTAB:   TST      (R3)+
427 000622              RETURN
428
429      ; COMPLEMENT OF PRIORITY
430
431
432 000624 005767 000352'      PRI:   TST      FLAGS      ; IS THIS A MUX EXPANSION ?
433 000630 001410              BEQ      10$          ; IF EQ, NO .. NO UPDATE
434 000632 126467 000013 000350'      CMPB   L.UNT(R4),INDEX ; IS THIS THE RIGHT UNIT ?
435 000640 001004              BNE      10$          ; IF NE, NO
436 000642 012713 000000G      MOV      #PR7,(R3)      ; GET COMPLEMENT OF PRIORITY 7
437 000646 046713 000222'      BIC      PRPRI,(R3)      ; ...
438 000652 005723      10$:    TST      (R3)+      ; SKIP PRIORITY
439 000654              RETURN
440
441      ; POINTER TO ALLOCATED SCOM
442
443
444 000656 062705 000004      SCOM:  ADD      #4,R5
445 000662 005723              TST      (R3)+
446 000664              RETURN
447
448      ; SECONDARY CSR
449
450

```

TMPEX7 - NTL TEMPLATE EXPANSION MACRO V05.03b Saturday 29-Jun-85 ^{K 7}00:44 Page 12-3
Symbol table

Operating system: RSX-11M/PLUS

Elapsed time: 00:00:18.59

SY:TMPEX7.V2,[132,134]TMPEX7/CR/~SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]TMPEX7

A\$\$CHK= 000000	K.CSR 000002	L\$\$ASG= 000000	N\$\$PEM= 000001	\$ADDR = ***** GX	
A\$\$CPS= 000000	K.PRI 000000	L\$\$DRV= 000000	P\$\$P45= 000000	\$AZFS = ***** GX	
A\$\$PRI= 000000	K.VCT 000001	L\$\$P11= 000001	P\$\$WRD= 000000	\$BIN = ***** GX	
A\$\$TRP= 000000	LD\$LP = 000000	L\$\$11R= 000000	Q\$\$OPT= 000010	\$CLIOS= ***** GX	
C\$\$CKP= 000000	LF.ACT= 100000	L.COST 000015	RTSPC 000242R	\$CLQUE= ***** GX	
C\$\$ORE= 000400	LF.BRO= 000400	L.CTL 000012	R\$\$DER= 000000	\$DFN = ***** GX	
C\$\$RSH= 177564	LF.BWL= 000007	L.CVA 177776	R\$\$K11= 000001	\$ERRWZ 000042R	002
D\$\$BUG= 177514	LF.ENA= 002000	L.DDM 000002	R\$\$SND= 000000	\$ERR8 000004R	002
D\$\$ISK= 000000	LF.LPB= 001000	L.DDS 000004	R\$\$11M= 000000	\$FLAGS= ***** GX	
D\$\$L11= 000001	LF.MDC= 000100	L.DLC 000003	SF.ACT= 000200	\$FLAG1= ***** GX	
D\$\$YNC= 000000	LF.MFL= 004000	L.DLM 000006	SF.ENA= 000100	\$FNOUT= ***** GX	
D\$\$YNM= 000000	LF.MTP= 000020	L.DLS 000010	SF.LPB= 000004	\$ICBLN= ***** GX	
E\$\$XPR= 000000	LF.PAC= 000200	L.FLG 000000	SF.MFL= 000040	\$LEFT = ***** GX	
FL.CHA= ***** GX	LF.RDY= 040000	L.KRBA 000016	SF.PAC= 000020	\$MXADD= ***** GX	
FL.DDM= ***** GX	LF.REA= 010000	L.LEN = 000022	SF.REA= 000010	\$MXSIZ= ***** GX	
FL.LLC= ***** GX	LF.SER= 000040	L.MPF 000022	SF.SER= 000001	\$NALL = ***** GX	
FMT0 000072R	LF.TIM= 000010	L.NMST 000020	SF.SVC= 000002	\$NAME = ***** GX	
FMT3 000073R	LF.UNL= 020000	L.NSTA 000014	SF.UNL= 000040	\$NVECT= ***** GX	
FM.0 = 000000	LF.X2P= 000000	L.OWNR 000021	S\$\$WRG= 000000	\$OFF = ***** GX	
FM.3 = 000000	LN.CLO= 000000	L.UNT 000013	S\$\$YSZ= 007600	\$PDVA = ***** GX	
FNDADD 000274R	LN.DUM= 000005	M\$\$CRB= 000124	S.COST 000001	\$PRV.N= ***** GX	
F\$\$LVL= 000001	LN.LOA= 000004	M\$\$CRX= 000000	S.FLG 000000	\$PTR = ***** GX	
F1.UMR= ***** GX	LN.LOO= 000003	M\$\$FCS= 000000	S.LEN 000004	\$SIZE = ***** GX	
G\$\$TPP= 000000	LN.OAU= 000001	M\$\$MGE= 000000	S.NMST 000002	\$SLTA = ***** GX	
G\$\$TSS= 000000	LN.OFF= 000001	M\$\$NET= 000000	S.OWNR 000003	\$SZPTR= ***** GX	
G\$\$TTK= 000000	LN.ON = 000000	M\$\$OVR= 000000	TMEXT 000000R	\$TM.IN 000000RG	002
G\$\$WRD= 000000	LN.OOP= 000004	N\$\$ACC= 000001	T\$\$KMG= 000000	\$TWRN= ***** GX	
I\$\$RAR= 000000	LN.OPE= 000001	N\$\$BUF= 000001	T\$\$MIN= 000000	\$TWE = ***** GX	
I\$\$RDN= 000000	LN.REF= 000002	N\$\$LDV= 000001	V\$\$CTR= 001000	\$XBUF2= ***** GX	
K\$\$CNT= 177546	LN.SER= 000002	N\$\$MCP= 000001	X\$\$DBT= 000000	\$XLEN2= ***** GX	
K\$\$CSR= 177546	LN.STA= 000017	N\$\$MLL= 000001	ZF.PSE= ***** GX	\$\$\$CSR = ***** GX	
K\$\$LDC= 000000	LN.SUB= 000360	N\$\$MOV= 000010	Z.DAI = ***** GX	\$SVECT= ***** GX	
K\$\$TPS= 000074	LN.TRI= 000006	N\$\$NCT= 000001	Z.FLG = ***** GX		

. ABS. 177776 000 (RW,I,GBL,ABS,OVR)
 000336 001 (RW,I,LCL,REL,CON)
 DATA 000120 002 (RW,D,LCL,REL,CON)
 Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 12057 Words (48 Pages)
 Size of core pool: 14440 Words (55 Pages)
 Operating system: RSX-11M/PLUS

Elapsed time: 00:00:10.33
 SY:TMPINI.V2,[132,134]TMPINI/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]TMPINI

```

174 .SBTTL .BIN,.BLKW,.BLKB
175
176
177 : ".BIN"
178 :
179 STATES$ BIN
180 TRANS$ !FNAME1,BIN1..ABFI
181 TRANS$ !ERROR,$EXIT
182 STATES$ BIN1
183 TRANS$ !END,$EXIT,$BINF
184 TRANS$ <','>
185 STATES$
186 .IF DF M$$MGE
187 TRANS$ !POSEXP,BIN2,$CORE2
188 .IFF
189 TRANS$ !EAT,ONLY2,$BINF
190 .IFTF
191 TRANS$ !ERROR,$EXIT
192 .IFT
193 STATES$ BIN2
194 TRANS$ $LAMDA,ONLY2,$BINF
195 .ENDC
196
197 : ".BLKB"
198 :
199 STATES$ BLKB
200 TRANS$ !POSEXP,ONLY1,$BLKB
201 TRANS$ !ERROR,$EXIT
202
203
204 : ".BLKW"
205 :
206 STATES$ BLKW
207 TRANS$ !POSEXP,ONLY1,$BLKW
208 TRANS$ !ERROR,$EXIT
209
210

```

TMPVCT - NTL VECTOR SETUP ROUTI MACRO V05.03b Saturday 29-Jun-85 00:47 Page 4

```

      LLL          SSSSSSSS   TTTTTTTTTT
      LL           SSSSSSSS   TTTT'TTTTT'
      LL              SS       TT
      LL             SS        TT
      LL            SS         TT
      LL           SS          TT
      LL          SSSSSS      TT
      LL         SSSSSS      TT
      LL                SS    TT
      LL               SS     TT
      LL              SS      TT
      LL             SS       TT
      LL            SS        TT
      LL           SSSSSSSS   TT
      LLLLLLLLLL  SSSSSSSS   TT

```

TMPVCT CREATED BY MACRO ON 29-JUN-85 AT 00:47 PAGE 1 K 11
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
AB	= ***** GX	7-377 7-381
FL.CHA	= ***** GX	6-137 7-344
FL.DDM	= ***** GX	6-135 7-354 7-363
FL.DLC	= ***** GX	7-348 7-365
FL.MUX	= ***** GX	7-370
FMKTMP	000024 R	#5-90
FMT1	000062 R	5-96 #5-108
FM.1	= 000000	#5-96
F1.VSE	= ***** GX	6-270 6-313
IS\$AS	= *****	6-159
K6TMP	000020 R	#5-88
LF.ACT	= 100000	#5-67
LF.BRO	= 000400	#5-67
LF.BWT	= 000007	#5-67
LF.ENA	= 002000	#5-67
LF.LPB	= 001000	#5-67
LF.MDC	= 000100	#5-67
LF.MFL	= 004000	#5-67
LF.MTP	= 000020	#5-67
LF.PAC	= 000200	#5-67
LF.RDY	= 040000	#5-67 8-432
LF.REA	= 010000	#5-67
LF.SER	= 000040	#5-67
LF.TIM	= 000010	#5-67
LF.UNL	= 020000	#5-67
LF.X2P	= 000000	#5-67
LLCTA	= ***** GX	9-471
LN.CLO	= 000000	#5-67
LN.DUM	= 000005	#5-67
LN.LOA	= 000004	#5-67
LN.LOO	= 000003	#5-67
LN.OAU	= 000003	#5-67
LN.OFF	= 000001	#5-67 9-487 9-497
LN.ON	= 000000	#5-67
LN.OOP	= 000004	#5-67
LN.OPE	= 000001	#5-67
LN.REF	= 000002	#5-67
LN.SER	= 000002	#5-67
LN.STA	= 000017	#5-67
LN.SUB	= 000360	#5-67
LN.TRI	= 000006	#5-67
L.COST	000015	#5-67
L.CTL	000012	#5-67 7-387 7-387
L.CVA	177776	#5-67
L.DDM	000002	#5-67 7-385 7-385 7-390 7-402
L.DDS	000004	#5-67 *7-389 *7-399 8-427
L.DLC	000003	#5-67 7-390 7-402
L.DLM	000006	#5-67 *7-357
L.DLS	000010	#5-67 *7-392 *7-404 8-429
L.FLG	000000	#5-67 *8-432
L.KRBA	000016	#5-67
L.LEN	= 000022	#5-67

UNLHC CREATED BY MACRO ON 29-JUN-85 AT 00:48 PAGE 3 K 12
MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

ADJDF\$	#5-47	5-49					
ASR\$	#5-47	6-81	6-96	6-106	6-114		
CALL	6-78	6-84	6-89	6-99	6-103	6-109	6-117
CTRDF\$	#5-47	5-52					
DHBDF\$	#5-47	5-50					
RETURN	6-119						
XPDDB\$	#5-47	5-51					

Symbol table

A\$\$CHK= 000000	G\$\$TSS= 000000	L\$\$GDN= 000004	M\$\$NET= 000000	TC\$LNG= 000002
A\$\$CPS= 000000	G\$\$TJK= 000000	L\$\$OFF= 000000	M\$\$OVR= 000000	TC\$TDA= 000001
A\$\$PRI= 000000	G\$\$WRD= 000000	L\$\$STR= 000002	N\$\$ACC= 000001	TR\$DCE= 000010
A\$\$TRP= 000000	HF\$DLM= 000002	L\$\$UP = 000003	N\$\$BUF= 000001	TR\$FIL= 000001
CF\$BLK= 000102	HF\$GWY= 000010	L\$\$WT = 000001	N\$\$LDV= 000001	TR\$OVR= 000004
CS\$DTE 000012	HF\$HOS= 000004	L\$ACHN 000012	N\$\$MCP= 000001	TR\$TRU= 000020
CS\$FLG 000015	HF\$XDF= 000020	L\$APVC 000014	N\$\$MLL= 000001	TR\$TX = 000002
CS\$GNAM 000016	HS\$CUG 000010	L\$ASVC 000010	N\$\$MOV= 000010	TS\$.BLK= 171700
CS\$LEN 000010	HS\$DST 000012	L\$AUC 000042	N\$\$NCT= 000001	TS\$.CKP= 000200
CS\$EN 000016	HS\$D29 000014	L\$CHLS 000034	N\$\$PEM= 000001	TS\$.CKR= 000100
CS\$LNK 000000	HS\$FLG 000000	L\$CHTB 000030	PSIPT = ***** GX	TS\$.EXE= 100000
CS\$NAM 000002	HS\$GLEN 000104	L\$CLEN= 000044	PS\$P45= 000000	TS\$.HLD= 002000
CS\$NML 000002	HS\$GLT 000044	L\$CTEN 000032	PS\$WRD= 000000	TS\$.MSG= 020000
CS\$PORT 000014	HS\$GNAM 000050	L\$CTIM 000040	Q\$ACTC 000004	TS\$.NRP= 010000
CS\$CKP= 000000	HS\$GNML= 000020	L\$DTEA 000020	Q\$AUC 000006	TS\$.OUT= 000400
CS\$ORE= 000400	HS\$GPT 000046	L\$DTEL 000017	Q\$CLEN= 000006	TS\$.RDN= 040000
CS\$RSH= 177564	HS\$HITS 000034	L\$GLEN 000134	Q\$CTIM 000000	TS\$.RUN= 004000
DEADST 000150R	HS\$HLEN 000044	L\$LEN 000122	Q\$ICRE 000014	TS\$.STP= 001000
DEALST= ***** GX	HS\$LBDA 000070	L\$LLCH 000016	Q\$LEN 000016	TS\$LAB= 000002
D\$ACL 000027	HS\$BDN 000072	L\$LNK 000000	Q\$MXAC 000012	TS\$LAB= 000000
D\$CUGL 000023	HS\$LDTE 000002	L\$MCHN 000036	Q\$MXVC 000002	TS\$CBUF 000004
D\$CUGN 000022	HS\$LEN 000042	L\$NETW 000114	Q\$TCLZ 000010	TS\$CEND 000006
D\$DATL 000030	HS\$LOTS 000032	L\$NJRE 000072	Q\$SOPT= 000010	TS\$CFLG 000000
D\$DSL 000022	HS\$NETW 000024	L\$NLRE 000104	R\$DAL 000013	TS\$CIN 000012
D\$DST 000010	HS\$NML = 000006	L\$NMAC 000076	R\$DTE 000014	TS\$CLEN 000020
D\$DTEL 000031	HS\$NPT 000022	L\$NMAS 000074	R\$LEN 000024	TS\$CMAP 000002
D\$FLEN 000032	HS\$PTB 000020	L\$NMST 000003	R\$LNK 000000	TS\$COUT 000014
D\$GLEN 000032	HS\$PVC 000006	L\$NNRE 000106	R\$NAM 000002	TS\$CSIZ 000010
D\$GVBL 000032	HS\$RDTE 000004	L\$NRBY 000046	R\$QWN 000010	TS\$CTCB 000016
D\$LNK 000000	HS\$RNW 000042	L\$NRCA 000066	R\$R0 = 000002	TS\$CUSE 000001
D\$NAM 000004	HS\$SVC 000036	L\$NRFS 000100	R\$R1 = 000004	TS\$CHK 000076
D\$NOD 000010	HS\$TRB 000016	L\$NRPK 000056	R\$R2 = 000006	TS\$FLG 000034
D\$OBJ 000003	HS\$XAVL 000100	L\$NRRE 000105	R\$R3 = 000010	TS\$LDN 000030
D\$PAL 000026	HS\$XBIA 000074	L\$NRST 000107	R\$R4 = 000012	TS\$LEN 000100
D\$PRI 000002	HS\$X29C 000040	L\$NTBY 000052	R\$R5 = 000014	TS\$TIM 000004
D\$PRL 000024	I\$SRAR= 000000	L\$NTCA 000070	R\$SDER= 000000	TS\$TRA 000000
D\$SHI 000020	I\$SRDN= 000000	L\$NTFS 000102	R\$SK11= 000001	TS\$TSK 000024
D\$SLO 000016	KSAR5 = ***** GX	L\$NTPK 000062	R\$SSND= 000000	TS\$RCNT 000002
D\$USL 000025	KS\$HIC 000004	L\$OMST 000002	R\$S11M= 000000	TS\$RFLG 000000
D\$VBL 000032	KS\$LEN 000006	L\$PSC 000112	SF\$BLK= 000001	TS\$RFRM 000007
D\$VBUG= 177514	KS\$LNK 000000	L\$PLS 000110	SF\$INC= 000002	TS\$RLIN 000004
D\$VSK= 000000	KS\$LOC 000002	L\$QCNT 000122	S\$ADSZ 000007	TS\$RSTS 000005
D\$VLL1= 000001	KS\$CNT= 177546	L\$QUEH 000124	S\$CTL 000002	TS\$RTYP 000001
D\$VYNC= 000000	KS\$CSR= 177546	L\$QUET 000130	S\$DTE 000020	TS\$SBUF 000001
D\$VYNM= 000000	KS\$LDC= 000000	L\$RTRY 000007	S\$FLG 000010	TS\$SDCT 000030
E\$XPR= 000000	KS\$TPS= 000074	L\$SLN 000004	S\$LEN 000032	TS\$DDM 000026
F\$LVL= 000001	LD\$LP = 000000	L\$ST 000005	S\$LFT 000012	TS\$DEV 000002
GF\$BUG= 000001	LN\$OFF= 000001	L\$TCLZ 000044	S\$LNK 000000	TS\$DUN 000031
GS\$CUG 000010	LN\$ON = 000000	L\$TIM 000006	S\$OWNR 000030	TS\$SNC 000000
GS\$DTE 000012	LN\$SHU= 000002	L\$SASG= 000000	S\$PKSZ 000004	TS\$SNM 000007
GS\$FLG 000014	LP\$EIP= 002000	L\$SDRV= 000000	S\$RCL 000013	TS\$STP 000020
GS\$GNAM 000016	LP\$ENB= 004000	L\$SP11= 000001	S\$STA 000003	TS\$GRP 000006
GS\$LEN 000016	LP\$PCT= 001400	L\$SP11R= 000000	S\$TIM 000014	TS\$LEN 000032
GS\$LNK 000000	LP\$TMR= 100000	M\$SCRB= 000124	S\$TMR 000016	TS\$MEM 000005
GS\$NAM 000002	LP\$UP = 010000	M\$SCRX= 000000	S\$WND 000006	TS\$XND= 000023
GS\$NML 000015	LP\$WTD= 020000	M\$SFCS= 000000	S\$WRG= 000000	TS\$TID 000023
GS\$TPP= 000000	LP\$WTS= 040000	M\$SMGE= 000000	S\$YSZ= 007600	TS\$TIU 000025

```

181                                     .SBTTL  XKRB - DE-ALLOCATE KRB
182
183                                     ;+
184                                     :
185                                     : XKRB - DE-ALLOCATE KRB
186                                     :
187                                     : INPUTS:
188                                     :   R0 - KRB ADDRESS
189                                     :
190                                     : OUTPUTS:
191                                     :   KRB IS DE-ALLOCATED
192                                     :
193                                     :-
194
195 000056                               XKRB::
196                                     .IF DF R$$11M
197                                     .IF DF R$$MPL
198                                     ADD  #K.PRM,R0      ; POINT TO START OF KRB
199                                     .ENDC
200                                     MOV   #KRBLEN,R1    ; GET LENGTH OF KRB ALLOCATION
201 000056 012701 000000G               SWSTK$ 10$        ;: ENTER KERNEL MODE
202 000062                                     CALLR $DECEX  ;: DE-ALLOCATE KRB
203 000066 10$:                          RETURN          ;: RETURN TO CALLER
204 000072
205                                     .IFF ; DF R$$11M
206
207                                     MOV   R3,-(SP)      ; GET 2 FREE REGISTERS
208                                     MOV   R4,-(SP)      ;
209                                     MOV   R0,R1         ; COPY ADDRESS OF BLOCK TO DEALLOCATE
210                                     MOV   #KRBLEN,R3    ; GET LENGTH OF THE BLOCK
211                                     .INHO               ; INHIBIT TASK SWITCHING
212                                     CALL  $$RLCB        ;: RELEASE THE BLOCK
213                                     .ENBO               ;: ENABLE TASK SWITCHING
214                                     MOV   (SP)+,R4      ; RESTORE REGISTERS
215                                     MOV   (SP)+,R3      ; ...
216                                     RETURN
217
218                                     .ENDC ; DF R$$11M
219

```

XLBR - RELEASE LIBRARY DESCRIPT MACRO V05.03b Saturday 29-Jun-85 00:49 Page 6-2
Symbol table

A\$\$CHK= 000000	F\$\$LVL= 000001	LBSIZ = 000000R	002 M\$\$OVR= 000000	002 R\$\$DER= 000000
A\$\$CPS= 000000	G\$\$TPP= 000000	LB.ADR= ***** GX	NLBLK 000006R	002 R\$\$K11= 000001
A\$\$PRI= 000000	G\$\$TSS= 000000	LB.SIZ= ***** GX	NLTPT = ***** GX	R\$\$SND= 000000
A\$\$TRP= 000000	G\$\$TIK= 000000	LB.USE= ***** GX	N\$\$ACC= 000001	R\$\$11M= 000000
BLKSIZ= 000001	G\$\$WRD= 000000	LD\$LP = 000000	N\$\$BUF= 000001	S\$\$WRG= 000000
C\$\$CKP= 000000	I\$\$RAR= 000000	L\$\$ASG= 000000	N\$\$LDV= 000001	S\$\$YSZ= 007600
C\$\$ORE= 000400	I\$\$RDN= 000000	L\$\$DRV= 000000	N\$\$MCP= 000001	T\$\$KMG= 000000
C\$\$RSH= 177564	K\$\$CNT= 177546	L\$\$P11= 000001	N\$\$MLL= 000001	T\$\$MIN= 000000
D\$\$BUG= 177514	K\$\$CSR= 177546	L\$\$11R= 000000	N\$\$MOV= 000010	V\$\$CTR= 001000
D\$\$ISK= 000000	K\$\$LDC= 000000	M\$\$CRB= 000124	N\$\$NCT= 000001	X\$\$DBT= 000000
D\$\$L11= 000001	K\$\$TPS= 000074	M\$\$CRX= 000000	N\$\$PEM= 000001	\$DEA18= ***** GX
D\$\$YNC= 000000	LBADR 000000R	002 M\$\$FCS= 000000	P\$\$P45= 000000	\$MAPX = ***** GX
D\$\$YNM= 000000	LBALL 000002R	002 M\$\$MGE= 000000	P\$\$WRD= 000000	\$XLBR 000000RG
E\$\$XPR= 000000	LBIAS 000004R	002 M\$\$NET= 000000	Q\$\$OPT= 000010	.CXLBR= ***** GX

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
000212 001 (RW,I,LCL,REL,CON)
DATA 000010 002 (RW,D,LCL,REL,CON)
Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 8942 Words (35 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:06.14
SY:XLBR.V2,[132,134]XLBR/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]XLBR

```

73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89 000000
90
91 000000 010046
92 000002 012700 000004'
93 000006 066700 001134'
94 000012 046700 001134'
95 000016 010067 000000'
96 000022 010067 001136'
97 000026 012746 001134'
98 000032 160016
99 000034 011667 000002'
100 000040 005020
101 000042 012610
102 000044 012600
103 000046
104
105 000001

;+
; $XXINI - INITIALIZE TASK POOL
;
; THIS ROUTINE IS CALLED TO DETERMINE THE STARTING ADDRESS AND LENGTH
; OF THE TASK POOL AND TO FORMAT THE POOL SO THAT IT MAY BE USED WITH
; THE "CORAL" ROUTINES.
;
; INPUTS:
;     NONE
;
; OUTPUTS:
;     $XXBUF = STARTING ADDRESS
;     $XXLEN = LENGTH
;     $XXAVL = "CORAL" POINTER
;-

.PSECT
$XXINI: MOV    R0, -(SP)           ;SAVE R0
        MOV    #XXBUF, R0        ;POINT AT THE BUFFER AREA
        ADD    $XXAVL-2, R0       ;ADD IN THE ROUNDING FACTOR
        BIC    $XXAVL-2, R0       ;ROUND UP, USUALLY TO A 4 BYTE BOUNDARY
        MOV    R0, $XXBUF         ;SET THE BUFFER ADDRESS
        MOV    R0, $XXAVL        ;SET THE "CORAL" POINTER ALSO
        MOV    #XXBUF+$BFLEN, -(SP) ;ADDRESS PAST END OF BUFFER
        SUB    R0, (SP)          ;GET REAL LENGTH OF BUFFER
        MOV    (SP), $XXLEN      ;SET THE BUFFER LENGTH
        CLR    (R0)+             ;CLEAR LINK TO NEXT SEGMENT
        MOV    (SP)+, (R0)       ;SET LENGTH OF SEGMENT
        MOV    (SP)+, R0         ;RECOVER R0
        RETURN                   ;FINISHED

.END

```

.TITLE IMPEX5 - NTL LINE TABLE UPDATE EXPANSION ROUTINES
 .IDENT /V05.00/

;**-1

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
 DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
 ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
 INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
 COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
 OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
 TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
 AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
 CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
 SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

NTL - MUX CHARACTERISTICS UPDATE EXPANSION ROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
 VERSION 2.0 RELEASE
- 2.00 14-DEC-79
 DECNET-11M/S V3.0
 DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
 DECNET-11M V3.1
 DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
 DECNET-11M V4.0
 DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
 DECnet-11M/S V4.2
 DECnet-11M-Plus V3.0
 DECnet-Micro/R SX V1.0

```

638
639 001246 011211      MOV      (R2),(R1)      ; * MAP TO THE NEXT UNLOAD BLOCK
640 001250 011202      MOV      (R2),R2      ; * GET NEXT UNLOAD BLOCK ADDRESS
641
642 001252 001327      BNE      10$          ; * IF NE, THERE IS ANOTHER ONE
643 001254 000416      BR       100$         ; *
644 001256 005742      80$:    TST      -(R2)      ; * RESET R2 TO FIRST WORD OF ENTRY
645 001260 010267 000160'  MOV      R2,UBA      ; * SAVE UNLOAD ENTRY ADDRESS
646
647 001264 011167 000156'  MOV      (R1),UBIAS      ; * AND BIAS
648 001270 042702 000077'  BIC      #77,R2      ; * RESET R2 TO START OF BLOCK
649 001274 012700 000162'  MOV      #BLOCK,R0      ; * MAKE A LOCAL COPY OF THE UNLOAD BLOCK
650 001300 012701 000040'  MOV      #32,R1      ; *
651 001304 012220      90$:    MOV      (R2)+(R0)+      ; *
652 001306      SOB      R1,90$      ; *
653
654 001312      100$:    RETURN      ; * BACK TO USER MODE AND THEN TO CALLER

```

```

LOCAL DATA

80          .SBTTL  LOCAL DATA
81
82          ***
83          : LOCAL DATA
84          ***
85
86          .PSECT  DATA,D
87
88
89          000000 000000          TEMP:  .WORD  0          ; TEMP LOCATION
90
91          :
92          : TRANSFER TABLE USING FUNCTION CODE AS INDEX
93          :
94          000002          TABLE:  TTAB$  ADDR$ ,NULL          ; EXPANSION ROUTINE IS FIRST ENTRY
95          000004          TTAB$  ADDRW ,NULL          ; CLEANUP ROUTINE IS SECOND ENTRY
96          000006          TTAB$  BLKB ,POP1
97          000010          TTAB$  BLKW ,POP1
98          000012          TTAB$  BYTE ,POP1
99          000014          TTAB$  CORE ,XCORE
100         000016          TTAB$  CSR ,POP1
101         000020          TTAB$  CTIM ,NULL
102         000022          TTAB$  DVCHA ,NULL
103         000024          TTAB$  EOF ,NULL
104         000026          TTAB$  FILE ,POP1
105         000030          TTAB$  INT ,POP2
106         000032          TTAB$  LIBR ,XLIBR
107         000034          TTAB$  LINKS ,NULL
108         000036          TTAB$  LFILE ,POP3
109         000040          TTAB$  LSTHD ,NULL
110         000042          TTAB$  LTAB ,NULL
111         000044          TTAB$  MPTAB ,POP3
112         000046          TTAB$  MXTAB ,POP1
113         000050          TTAB$  NOD ,NULL
114         000052          TTAB$  PRI ,NULL
115         000054          TTAB$  PRTOV ,POP1
116         000056          TTAB$  SCOM ,XSCOM
117         000060          TTAB$  SECSR ,NULL
118         000062          TTAB$  SLNB ,NULL
119         000064          TTAB$  SLNW ,NULL
120         000066          TTAB$  STNB ,NULL
121         000070          TTAB$  STNW ,NULL
122         000072          TTAB$  UNB ,NULL
123         000074          TTAB$  UNW ,NULL
124         000076          TTAB$  WORD ,POP2
125         000100          TTAB$  APR ,POP1
126         000102          TTAB$  DPRB ,POP1
127         000104          TTAB$  DPRW ,POP2
128         000106          TTAB$  TIME ,POP2
129         000110          TTAB$  MXPTB ,MXPTB
130         000112          TTAB$  PECHA ,NULL
131         000114          TTAB$  X3CH01 ,NULL
132         000116          TTAB$  X3CH02 ,NULL
133         000120          TTAB$  X3CH03 ,NULL
134         000122          TTAB$  X3CH04 ,NULL
135         000124          TTAB$  X3CH05 ,NULL
136         000126          TTAB$  X3CH06 ,NULL

```

LOCAL DATA

Z10 001400 046724 000344'
Z11 001404

BIC PRPRI,(R4)+ ; CLEAR ACTUAL PRIORITY BITS
RETURN

TMPEX6 CREATED BY MACRO ON 29-JUN-85 AT 00:43 PAGE 4 L 5
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
STNW	000704 R	6-121 #8-493
S.COST	000001	#5-58
S.FLG	000000	#5-58
S.LEN	000004	#5-58
S.NMST	000002	#5-58
S.OWNR	000003	#5-58
TABLE	000002 R	#6-94 7-289 12-725 12-750 14-829
TEMP	000000 R	#6-89 *14-848 *14-849 14-850 *14-851 14-852
TIME	002110 R	6-128 #14-848
UNB	000746 R	6-122 #8-508
UNW	000770 R	6-123 #8-518
WORD	001012 R	6-124 #8-528
X2CH01	002306 R	6-142 #16-942
X2CH02	002334 R	6-143 #16-952
X2CH03	002362 R	6-144 #16-962
X2CH04	002410 R	6-145 #16-972
X2CH05	002436 R	6-146 #16-982
X3CH01	002204 R	6-131 #15-885
X3CH02	002212 R	6-132 #15-890
X3CH03	002220 R	6-133 #15-895
X3CH04	002226 R	6-134 #15-900
X3CH05	002234 R	6-135 #15-905
X3CH06	002242 R	6-136 #15-910
X3CH07	002250 R	6-137 #15-915
X3CH08	002256 R	6-138 #15-920
X3CH09	002264 R	6-139 #15-925
X3CH10	002272 R	6-140 #15-930
X3CH11	002300 R	6-141 #15-935
ZTIME	= ***** GX	14-848
\$ADDR	= ***** GX	7-277 11-612 12-752
\$BIAS	= ***** GX	7-275
\$BIN	= ***** GX	7-282
\$FLAGS	= ***** GX	7-271 16-943 16-953 16-963 16-973 16-983
\$ICBAD	= ***** CX	7-263 11-617
\$ICBLN	= ***** GX	7-264
\$JSROF	= ***** GX	11-614
\$LDA	= ***** GX	13-797 *13-801
\$MAPX	= ***** GX	7-276
\$MXSIZ	= ***** GX	14-841
\$PDVA	= ***** GX	7-266
\$PTR	= ***** GX	7-283
\$SLTSV	= ***** GX	8-370
\$SMTSZ	= ***** GX	7-265
\$TMEX6	000000 RG	#7-258
\$UBIAS	= ***** GX	9-546 10-587 13-796
\$UDA	= ***** GX	9-547 *9-553 10-588 *10-591
\$VECT1	= ***** GX	11-613 *11-702
.EOF	= ***** GX	12-722 12-747 14-826

```

451 000666 005767 000352'   SECSR: TST   FLAGS   : IS IT MUX EXPANSION ?
452 000672 001420           BEQ    20$      : NO
453 000674 016700 000350'   MOV    INDEX,R0  : GET SECONDARY FILE COUNTER
454 000700 126400 000013    CMPB   L,UNT(R4),R0 : DO UNIT NUMBERS MATCH ?
455 000704 001013           BNE    20$      : NO
456 000706 032767 000000G 000346' BIT    #FL.KMX,PRFLAG : IS THIS A COMIOP MUX LINE ?
457 000714 001403           BEQ    10$      : NO
458 000716 006200           ASR     R0       : YES... DIVIDE COUNTER BY 8
459 000720 006200           ASR     R0       : (NUMBER OF LINES PER CONTROLLER)
460 000722 006200           ASR     R0
461 000724 006300 000154'   10$: ASL     R0       : CONVERT TO A WORD INDEX
462 000726 016023           MOV    PRSCSR(R0),(R3)+; COPY SECONDARY CSR ADDRESS
463 000732 000401           BR      30$
464 000734 005723           20$: TST   (R3)+
465 000736           30$: RETURN
466
467           ;
468           ; SYSTEM LINE NUMBER (BYTE)
469
470 000740 005203   SLNB: INC     R3
471 000742           RETURN
472
473           ;
474           ; SYSTEM LINE NUMBER (WORD)
475
476 000744 005723   SLNW: TST   (R3)+
477 000746           RETURN
478
479           ;
480           ; STATION NUMBER (BYTE)
481
482 000750 005203   STNB: INC     R3
483 000752           RETURN
484
485           ;
486           ; STATION NUMBER (WORD)
487
488 000754 005723   STNW: TST   (R3)+
489 000756           RETURN
490
491           ;
492           ; UNIT NUMBER (BYTE)
493
494 000760 005203   UNB: INC     R3
495 000762           RETURN
496
497           ;
498           ; UNIT NUMBER (WORD)
499
500 000764 005723   UNW: TST   (R3)+
501 000766           RETURN
502
503           ;
504           ; SINGLE WORD
505
506 000770 122525   WORD: CMPB   (R5)+,(R5)+
507 000772 005723   TST   (R3)+

```

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES							
ADDRB	000360 R	5-91	#7-293						
ADDRW	000364 R	5-92	#7-299						
APR	001346 R	5-122	#9-637						
BLKB	000370 R	5-93	#7-305						
BLKW	000406 R	5-94	#7-315						
BLOCK	000360 R	#5-179	6-257	10-713					
BYTE	000426 R	5-95	#7-326						
CORE	000432 R	5-96	#7-332						
CSR	000442 R	5-97	#7-340						
CTIM	000450 R	5-98	#7-346						
DPRB	001352 R	5-123	#9-644						
DPRW	001356 R	5-124	#9-650						
DVCHA	000454 R	5-99	#7-352						
EOF	000554 R	5-100	#7-381						
FILE	001002 R	5-101	#8-514						
FLAGS	000352 R	#5-172	*6-202	7-352	7-369	7-432	7-451	*8-513	8-517
		*8-574	*9-580	*9-616	12-783	12-796	12-809	12-822	12-835
FLT	001362 R	#10-667							*8-546
FLT16	001362 R	6-242	#10-666						
FL.KMX	= ***** GX	7-456							
INDEX	000350 R	#5-167	*6-201	7-354	7-371	7-434	7-453	8-519	*8-541
		*8-545	*8-569	8-570	*8-573	9-588	*9-610	9-611	*9-615
		12-798	12-811	12-824	12-837				8-542
		5-102	#7-386						12-785
INT	000556 R	6-269	8-522	9-591					
IS\$AS	= *****	5-105	#8-558						
LFILE	001112 R	#5-60							
LF.ACT	= 100000	#5-60							
LF.BRO	= 000400	#5-60							
LF.BWT	= 000007	#5-60							
LF.ENA	= 002000	#5-60							
LF.LPB	= 001000	#5-60							
LF.MDC	= 000100	#5-60							
LF.MFL	= 004000	#5-60							
LF.MTP	= 000020	#5-60							
LF.PAC	= 000200	#5-60							
LF.RDY	= 040000	#5-60							
LF.REA	= 010000	#5-60							
LF.SER	= 000040	#5-60							
LF.TIM	= 000010	#5-60							
LF.UNL	= 020000	#5-60							
LF.X2P	= 000000	#5-60							
LIBR	000566 R	5-103	#7-393						
LINKS	000576 R	5-104	#7-401						
LN.CLO	= 000000	#5-60							
LN.DUM	= 000005	#5-60							
LN.LOA	= 000004	#5-60							
LN.LOO	= 000003	#5-60							
LN.OAU	= 000003	#5-60							
LN.OFF	= 000001	#5-60							
LN.ON	= 000000	#5-60							
LN.OOP	= 000004	#5-60							
LN.OPE	= 000001	#5-60							

TMPINI CREATED BY MACRO ON 29-JUN-85 AT 00:44 PAGE 1 L 8
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
FL.CHA	= ***** GX	6-141
FL.DDM	= ***** GX	6-135
FL.LLC	= ***** GX	6-181
FMT0	000072 R	5-88 #5-97
FMT3	000073 R	5-87 #5-98
FM.O	= 000000	#5-88
FM.3	= 000000	#5-87
FNDADD	000274 R	6-155 #7-201
F1.UMR	= ***** GX	6-126
K.CSR	000002	6-166
K.VCT	000001	6-152
LF.ACT	= 100000	#5-59
LF.BRO	= 000400	#5-59
LF.BWT	= 000007	#5-59
LF.ENA	= 002000	#5-59
LF.LPB	= 001000	#5-59
LF.MDC	= 000100	#5-59
LF.MFL	= 004000	#5-59
LF.MTP	= 000020	#5-59
LF.PAC	= 000200	#5-59
LF.RDY	= 040000	#5-59
LF.REA	= 010000	#5-59
LF.SER	= 000040	#5-59
LF.TIM	= 000010	#5-59
LF.UNL	= 020000	#5-59
LF.X2F	= 000000	#5-59
LN.CLO	= 000000	#5-59
LN.DUM	= 000005	#5-59
LN.LOA	= 000004	#5-59
LN.LOD	= 000003	#5-59
LN.OAU	= 000003	#5-59
LN.OFF	= 000001	#5-59
LN.ON	= 000000	#5-59
LN.OOP	= 000004	#5-59
LN.OPE	= 000001	#5-59
LN.REF	= 000002	#5-59
LN.SER	= 000002	#5-59
LN.STA	= 000017	#5-59
LN.SUB	= 000360	#5-59
LN.TRI	= 000006	#5-59
L.COST	000015	#5-59
L.CTL	000012	#5-59
L.CVA	177776	#5-59
L.DDM	000002	#5-59
L.DDS	000004	#5-59 6-138
L.DLC	000003	#5-59
L.DLM	000006	#5-59
L.DLS	000010	#5-59
L.FLG	000000	#5-59
L.KRBA	= 000016	#5-59 6-142
L.LEN	= 000022	#5-59
L.MPF	000022	#5-59

```

212                                     .SBTTL .BYTE,.CORE
213                                     ;
214                                     ; ".BYTE"
215                                     ;
216 STATES BYTE
217 TRANS !EXP,BYTE1,$BYTE
218 TRANS <'>,BYTE2,$ZBYTE
219 TRANS !ERROR,$EXIT
220 STATES BYTE1
221 TRANS !END,$EXIT
222 TRANS <'>
223 STATES BYTE2
224 TRANS !EXP,BYTE1,$BYTE
225 TRANS <'>,BYTE2,$ZBYTE
226 TRANS !END,$EXIT,$ZBYTE
227
228                                     ;
229                                     ; ".CORE"
230                                     ;
231 STATES CORE
232 TRANS !POSEXP,CORE1,$CORE1
233 TRANS !ERROR,$EXIT
234 STATES CORE1
235 TRANS !END,$EXIT,$CORE
236 TRANS <'>
237 STATES
238 M$$MGE
239 TRANS !POSEXP,CORE2,$CORE2
240 TRANS
241 TRANS !EAT,ONLY2,$CORE
242 TRANS
243 TRANS !ERROR,$EXIT
244 TRANS
245 STATES CORE2
246 TRANS $LAMDA,ONLY2,$CORE
247 TRANS
248 TRANS

```

.TITLE TMPVCT - NTL VECTOR SETUP ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

NTL - INTERRUPT VECTOR SETUP ROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/Rsx V1.0

TMPVCT CREATED BY MACRO ON 29-JUN-85 AT 00:47 PAGE 2 L 11
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
L.MPF	000022	#5-67 9-479
L.NMST	000020	#5-67 *9-497
L.NSTA	000014	#5-67 9-472
L.OWNR	000021	#5-67 *9-496
L.UNT	000013	#5-67
MANINI	000642 R	8-431 #9-460
M\$MGE	= 000000	6-248 7-347
NSITAB	000000 R	#5-78 6-161
NSITMP	000022 R	#5-89
NS0	= ***** GX	5-78
NS1	= ***** GX	5-79
NS2	= ***** GX	5-80
NS3	= ***** GX	5-81
NS4	= ***** GX	5-82
NS5	= ***** GX	5-83
NS6	= ***** GX	5-84
NS7	= ***** GX	5-85
PR7	= ***** GX	6-228 6-261
PS	= ***** GX	*6-228 *6-268
R\$EIS	= *****	6-159
R\$EMPL	= *****	5-97 6-152 6-230 6-271 6-306 8-434
R\$11D	= *****	6-159
R\$11M	= 000000	6-149
SF.ACT	= 000200	#5-67
SF.ENA	= 000100	#5-67
SF.LPB	= 000004	#5-67
SF.MFL	= 000040	#5-67
SF.PAC	= 000020	#5-67
SF.REA	= 000010	#5-67
SF.SER	= 000001	#5-67
SF.SVC	= 000002	#5-67
SF.UNL	= 000040	#5-67
SLTMA	= ***** y	7-380 9-462 9-470
SLTMM	= ***** G.A.	7-376 9-461
S\$BAS	= * ***	5-96
S.COST	000001	#5-67
S.FLG	000000	#5-67
S.LEN	000004	#5-67 9-488
S.NMST	000002	#5-67 *9-487
S.OWNR	000003	#5-67 *9-486
ZF.PSE	= ***** GX	6-133
Z.DAT	= ***** GX	*7-368
Z.DSP	= ***** GX	7-357
Z.FLG	= ***** GX	6-133
Z.PCB	= ***** GX	8-424
\$ADDR	= ***** GX	6-249 6-262 7-351 7-360
\$DDMPV	= ***** GX	8-423
\$DLCPV	= ***** GX	7-353 8-419
\$ERR38	000026 R	#5-96 6-315
\$FLAGS	= ***** GX	6-135 7-344 7-348 7-354 7-363 7-365 7-370
\$FLAG1	= ***** GX	6-130 *6-270 6-313 7-340
\$ICBAD	= ***** GX	6-252

FILEID**UNLPSI

```

UU      UU      NN      NN      LL      PPPPPPPP      SSSSSSSS      IIIIII
UU      UU      NN      NN      LL      PPPPPPPP      SSSSSSSS      IIIIII
UU      UU      NN      NN      LL      PP          SS          II
UU      UU      NN      NN      LL      PP          PP          SS          II
UU      UU      NNNN      NN      LL      PP          PP          SS          II
UU      UU      NNNN      NN      LL      PP          PP          SS          II
UU      UU      NN      NN      LL      PPPPPPPP      SSSSSS      II
UU      UU      NN      NN      LL      PPPPPPPP      SSSSSS      II
UU      UU      NN      NN      LL      PP          SS          II
UU      UU      NN      NNNN      LL      PP          SS          II
UU      UU      NN      NNNN      LL      PP          SS          II
UU      UU      NN      NN      LL      PP          SS          II
UU      UU      NN      NN      LL      PP          SS          II
UUUUUUUU      NN      NN      LLLLLLLLLL      PP          SSSSSSSS      IIIIII
UUUUUUUUUU      NN      NN      LLLLLLLLLL      PP          SSSSSSSS      IIIIII

```

```

....
....
....

```

```

LL      SSSSSSSS      TTTTTTTTTT
LL      SSSSSSSS      TTTTTTTTTT
LL      SS          TT
LL      SS          TT
LL      SS          TT
LL      SS          TT
LL      SSSSSS      TT
LL      SSSSSS      TT
LL      SS          TT
LL      SS          TT
LL      SS          TT
LL      SS          TT
LL      SS          TT
LLLLLLLLLL      SSSSSSSS      TT
LLLLLLLLLL      SSSSSSSS      TT

```


T\$SUNT 000004	T.TIO 000057	T3.REM= 020000	XX29 000560R	X\$PS 000010
T\$KMG= 000000	T.TKSZ 000060	T3.ROV= 000040	X\$ABQ 000076	X\$RPR 000013
T\$MIN= 000000	T.TUCB 000026	T3.RST= 000400	X\$ALF 000055	X\$RPS 000012
T.ACTL 000052	T2.ABO= 000100	T3.SLV= 002000	X\$AUC 000056	X\$RTRY 000002
T.ASTL 000016	T2.AST= 100000	T3.SWS= 000002	X\$CLEN= 000025	X\$RXI 000104
T.ATT 000062	T2.CAF= 000400	V\$SCTR= 001000	X\$CTIM 000060	X\$RXQ 000072
T.CPCB 000004	T2.CHK= 020000	XAS\$NCO= 000020	X\$DIAG 000003	X\$SS 000004
T.DPRI 000040	T2.CKD= 010000	XAS\$RR= 000200	X\$DTE 000024	X\$ST 000005
T.EFLG 000022	T2.DST= 040000	XAS\$UCL= 000040	X\$FLG 000014	X\$TCLZ 000030
T.IOC 000003	T2.FXD= 002000	XAS\$URE= 000100	X\$GLEN 000106	X\$TIMC 000001
T.LBN 000041	T2.HLT= 000200	XAS\$NCO= 000020	X\$LCN 000026	X\$TIMR 000000
T.LDV 000044	T2.REX= 001000	XF\$OFF= 000200	X\$LEN 000106	X\$TXI 000102
T.LNK 000000	T2.SEF= 004000	XF\$ORI= 000002	X\$MDWN 000023	X\$TXQ 000066
T.MXSZ 000050	T2.SPN= 000004	XF\$OTI= 000001	X\$NLRE 000052	X\$TYP 000015
T.NAM 000006	T2.STP= 000020	XF\$TIL= 000010	X\$NNRE 000054	X\$USR 000022
T.OFF 000066	T2.WFR= 000001	XF\$TIR= 000004	X\$NPL 000006	X\$WAQ 000062
T.PCB 000046	T3.ACP= 100000	XLDTE 000250R	X\$NPR 000007	X\$WSZ 000020
T.PRI 000002	T3.CAL= 000100	XPORT 000436R	X\$NRBY 000032	X\$SDBT= 000000
T.RCVL 000012	T3.CLI= 001000	XTRA 000610R	X\$NRPK 000042	\$CEACX= ***** GX
T.RRFL 000072	T3.GFL= 000010	XT\$CHN= 000100	X\$NRRE 000053	\$DEA16= ***** GX
T.SAST 000054	T3.MCR= 004000	XT\$DGR= 000002	X\$NTBY 000036	\$DECEX= ***** GX
T.SRCT 000071	T3.NET= 000020	XT\$FAS= 000200	X\$NTPK 000046	X\$DEAC= ***** GX
T.STAT 000032	T3.NSD= 000200	XT\$INC= 000004	X\$PKSZ 000016	\$XPSI 000000RG
T.ST2 000034	T3.PMD= 040000	XT\$OUC= 000010	X\$PR 000011	\$\$\$ = 000062
T.ST3 000036	T3.PRV= 010000	XT\$PVC= 000001	X\$PRT 000021	\$\$\$R = 000010
T.TCBL 000030				

. ABS. 000134 000 (RW,I,GBL,ABS,OVR)
 000724 001 (RW,I,LCL,REL,CON)

Errors detected: 0

*** Assembler statistics

Work file reads: 60
 Work file writes: 63
 Size of work file: 20003 Words (79 Pages)
 Size of core pool: 17608 Words (67 Pages)
 Operating system: RSX-TIM/PLUS

Elapsed time: 00:00:50.12
 DB2:UNLPSI.[134,[132,134]UNLPSI/CR/-SP=DB2:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]UNLPSI

```

221                                     .SBTTL FNDADD - FIND LINE TABLE ADDRESS
222
223
224                                     +
225                                     FNDADD - FIND LINE TABLE ADDRESS
226
227                                     INPUTS:
228                                     R3 = VECTOR ADDRESS
229
230                                     OUTPUTS:
231                                     R3 = LINE TABLE ADDRESS
232
233                                     -
234
235 FNDADD::
236                                     .IF DF R$$MPL
237                                     SWSTK$ 20$                                     ;; ENTER KERNEL MODE ON M+ ONLY
238
239                                     .IFTF ; DF R$$MPL
240
241 000074 011303 MOV (R3),R3                                     ;; POINT TO ICB OR LINE TABLE
242 000076 010302 MOV R3,R2                                     ;; COPY ICB OR LINE TABLE ADDRESS
243 000100 066702 000000G ADD $ICBLN,R2                                     ;; POINT TO END IF ICB
244 000104 162702 000004 SUB #4,R2                                     ;; POINT TO POSSIBLE JSR INSTRUCTION
245 000110 022712 004537 CMP #4537,(R2)                                     ;; IF "JSR R5,X", THEN THIS IS LINE TABLE
246 000114 001406 BEQ 10$                                     ;; ..ADDRESS
247 000116 022712 000137 CMP #137,(R2)                                     ;; IF "JMP line-table", THEN LINE TABLE
248
249                                     .IFT ; DF R$$MPL
250
251 BEQ 5$                                     ;; ..ADDRESS IS NEXT WORD
252 CLR R3                                     ;; ELSE, ERROR
253 BR 10$                                     ;; ...
254
255                                     .IFF ; DF R$$MPL
256
257 000122 001004 BNE 101$                                     ; ..ADDRESS IS NEXT WORD, ELSE ERROR
258
259                                     .IFTF ; DF R$$MPL
260
261 000124 062702 000002 5$: ADD #2,R2                                     ;; POINT TO REAL TIME LINE TABLE ADDRESS
262 000130 011203 MOV (R2),R3                                     ;; SAVE REAL LINE TABLE ADDRESS
263
264 000132 10$: .IFT ; DF R$$MPL
265
266 MOV R3,10(SP)                                     ;; RETURN LINE TABLE ADDRESS IN R3
267 RETURN                                     ;; BACK TO USER MODE
268
269 20$: TST R3                                     ; SUCCESS ?
270 BEQ 101$                                     ; IF EQ, NO .. ERROR
271 .ENDC ; DF R$$MPL
272
273 000132 RETURN
274
275 ; ERROR CONDITIONS
276
277 000134 101$: CRASH                                     ; WHAT IS R3 POINTING AT ???

```

XLBR CREATED BY MACRO ON 29-JUN-85 AT 00:49 PAGE 1 L 15
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
BLKSIZ	= 000001	#5-75 6-140
LBADR	000000 R	#5-65 5-67 *6-92 6-100
LBALL	000002 R	#5-66 *6-93 *6-104 6-134
LBIAS	000004 R	#5-68 *6-94 *6-114 6-138
LBSIZ	= 000000 R	#5-67 *6-105 6-136
LB.ADR	= ***** GX	6-104
LB.SIZ	= ***** GX	6-105
LB.USE	= ***** GX	*6-101
M\$MGE	= 000000	5-64 5-75 5-76
NLBLK	000006 R	#5-69 *6-95 6-107 6-115
NTLPT	= ***** GX	6-95
\$DEA18	= ***** GX	6-137 6-141
\$MAPX	= ***** GX	6-99
\$XLBR	000000 RG	#6-90
.CXLBR	= ***** GX	6-117 *6-119 6-122

Symbol table

A\$\$CHK= 000000	F\$\$LVL= 000001	L\$\$DRV= 000000	N\$\$MCP= 000001	S\$\$WRG= 000000
A\$\$CPS= 000000	G\$\$TPP= 000000	L\$\$P11= 000001	N\$\$MLL= 000001	S\$\$YSZ= 007600
A\$\$PRI= 000000	G\$\$TSS= 000000	L\$\$11R= 000000	N\$\$MOV= 000010	T\$\$KMG= 000070
A\$\$TRP= 000000	G\$\$TTK= 000000	M\$\$CRB= 000124	N\$\$NCT= 000001	T\$\$MIN= 000000
C\$\$CKP= 000000	G\$\$WRD= 000000	M\$\$CRX= 000000	N\$\$PEM= 000001	V\$\$CTR= 001000
C\$\$ORE= 000400	I\$\$RAR= 000000	M\$\$FCS= 000000	P\$\$P45= 000000	XXBUF 000004R 002
C\$\$RSH= 177564	I\$\$RDN= 000000	M\$\$MGE= 000000	P\$\$WRD= 000000	X\$\$DBT= 000000
D\$\$BUG= 177514	K\$\$CNT= 177546	M\$\$NET= 000000	Q\$\$OPT= 000010	\$BFLN= 001130 G
D\$\$ISK= 000000	K\$\$CSR= 177546	M\$\$OVR= 000000	R\$\$DER= 000000	\$XXAVL 001136RG 002
D\$\$L11= 000001	K\$\$LDC= 000000	N\$\$ACC= 000001	R\$\$K11= 000001	\$XXBUF 000000RG 002
D\$\$YNC= 000000	K\$\$TPS= 000074	N\$\$BUF= 000001	R\$\$SND= 000000	\$XXINI 000000RG
D\$\$YNM= 000000	LD\$LP= 000000	N\$\$LDV= 000001	R\$\$11M= 000000	\$XXLEN 000002RG 002
E\$\$XPR= 000000	L\$\$ASG= 000000			

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
 000050 001 (RW,I,LCL,REL,CON)
 DATA 001142 002 (RW,D,LCL,REL,CON)
 Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 8926 Words (35 Pages)
 Size of core pool: 14440 Words (55 Pages)
 Operating system: RSX-11M/PLUS

Elapsed time: 00:00:05.08
 SY:XXBUF.V2,[132,134]XXBUF/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]XXBUF

```

55      .SBTTL  MACRO DEFINITIONS
56
57      :***
58      : MACRO CALLS
59      :***
60
61      .MCALL  NKRDf$,SLTDf$,SAVRG,RESRG
62
63      000000      SLTDf$      ; DEFINE SLT OFFSETS
64
65      .IF DF  R$$MPL
66      .MCALL  KRBDf$
67      KRBDf$      ; DEFINE KRB OFFSETS
68      .ENDC
69
70      000000      NKRDf$      ; DEFINE NETWORK KRB OFFSETS
71
72      :***
73      : LOCAL MACROS
74      :***
75
76      :
77      : DEFINE TRANSFER TABLE ENTRY
78      :
79      .MACRO  TTAB$  AA,BB
80      .WORD   AA
81      .ENDM   TTAB$
82
83      :***
84      : LOCAL DATA
85      :***
86
87      000000      .PSECT  DATA,D
88
89
90      :
91      : TRANSFER TABLE USING FUNCTION CODE AS INDEX
92      :
93      TABLE:  TTAB$  ADDR$,NULL      ; EXPANSION ROUTINE IS FIRST ENTRY
94               TTAB$  ADDR$,NULL      ; CLEANUP ROUTINE IS SECOND ENTRY
95               TTAB$  BLKB,POP1
96               TTAB$  BLKW,POP1
97               TTAB$  BYTE,POP1
98               TTAB$  CORE,XCORE
99               TTAB$  CSR,POP1
100              TTAB$  CTIM,NULL
101              TTAB$  DVCHA,NULL
102              TTAB$  EOF,NULL
103              TTAB$  FILE,POP1
104              TTAB$  INT,POP2
105              TTAB$  LIBR,XLIBR
106              TTAB$  LINK$,NULL
107              TTAB$  LFILE,POP3
108              TTAB$  LSTHD,NULL
109              TTAB$  LTAB,NULL
110              TTAB$  MPTAB,POP3
111              TTAB$  MXTAB,POP1
112              TTAB$  NOD,NULL
113              TTAB$  PRI,NULL

```

```

656                      .SBTTL X3CH
657
658                      ;
659                      ; DEFAULT BLOCK SIZE
660
661 001314 005723      X3CH01: TST (R3)+
662 001316                      RETURN
663
664                      ; MAXIMUM BLOCK SIZE
665
666 001320 005723      X3CH02: TST (R3)+
667 001322                      RETURN
668
669                      ; DEFAULT WINDOW SIZE
670
671 001324 005203      X3CH03: INC R3
672 001326                      RETURN
673
674                      ; MAXIMUM WINDOW SIZE
675
676 001330 005203      X3CH04: INC R3
677 001332                      RETURN
678
679                      ; CALL TIMER
680
681 001334 005203      X3CH05: INC R3
682 001336                      RETURN
683
684                      ; CLEAR TIMER
685
686 001340 005203      X3CH06: INC R3
687 001342                      RETURN
688
689                      ; RESET TIMER
690
691 001344 005203      X3CH07: INC R3
692 001346                      RETURN
693
694                      ; RESTART TIMER
695
696 001350 005203      X3CH08: INC R3
697 001352                      RETURN
698
699                      ; MAXIMUM CLEAR VALUE
700
701 001354 005203      X3CH09: INC R3
702 001356                      RETURN
703
704                      ; MAXIMUM RESETS VALUE
705
706 001360 005203      X3CH10: INC R3
707 001362                      RETURN
708
709                      ; MAXIMUM RESTARTS VALUE
710
711 001364 005203      X3CH11: INC R3
712 001366                      RETURN

```

```

137 000130          TTAB$  X3CH07,NULL
138 000132          TTAB$  X3CH08,NULL
139 000134          TTAB$  X3CH09,NULL
140 000136          TTAB$  X3CH10,NULL
141 000140          TTAB$  X3CH11,NULL
142 000142          TTAB$  X2CH01,NULL
143 000144          TTAB$  X2CH02,NULL
144 000146          TTAB$  X2CH03,NULL
145 000150          TTAB$  X2CH04,NULL
146 000152          TTAB$  X2CH05,NULL
147
148          .JF DF  R$511M
149
150          ; INITIAL DISPLACEMENT FOR ALL ADDRESS REQUESTS
151
152 000154 120000    PRAPR: .WORD 120000          ; USE APR5
153
154          .JFF
155
156          ; INITIAL DISPLACEMENT FOR ALL ADDRESS REQUESTS
157
158 000160 60000     PRAPR: .WORD 60000           ; USE APR3
159
160          .ENDC
161
162          ; SECONDARY FILE INDEX VALUE
163
164 000166          INDEX: .BLKW 1
165
166          ; MUX/MTP FLAGS
167
168 000170          FLAGS: .BLKW 1                ; 100000=MUX, 200=MTP
169
170          ; STATUS FLAG
171
172 000174          PRFLAG: .BLKW 1              ; COPY OF GLOBAL STATUS FLAG
173
174 000176          STLIN: .BLKW 1              ; START OF LINE TABLE
175
176          ; ICB ALLOCATION DATA
177
178 000180          ICBAD: .BLKW 1                ; ADDRESS OF ICB
179 000182          ICBPT: .BLKW 1              ; POINTER INTO ICB OF CURRENT ".INT"
180 000184          NXTLN: .BLKW 1              ; COUNT OF CURRENT ".INT" * 2
181 000186          ICBLN: .BLKW 1              ; ICB LENGTH
182
183          ; LOCAL COPY OF GLOBAL SYMBOL TABLE FROM "NTLR00"
184
185 000188          PRPDVA: .BLKW 1              ; PDV ADDRESS
186 000190          PRNAME: .BLKW 1             ; RAD50 PDV NAME
187 000192          PRBIAS: .BLKW 1            ; RELOCATION BIAS
188
189 000194          PRPCB: .BLKW 1              ; PCB ADDRESS
190
191
192
193

```

```

LOCAL DATA
713
714
715
716 001406 012767 100000 000160' MXTAB: MOV #100000,FLAGS
717 001414 112546 FILE: MOVB (R5)+,(SP)
718 001416 010546 MOV R5,-(SP)
719 001420 011605 10$: MOV (SP),R5
720 001422 005003 20$: CLR R3
721 001424 152503 BISB (R5)+,R3
722 001426 122703 000000G CMPB #.EOF,R3
723 001432 001404 BEQ 30$
724 001434 106303 ASLB R3
725 001436 CALL @TABLE-2(R3)
726 001442 000767 BR 20$
727 001444 005267 000156' 30$: INC INDEX
728 001450 126667 000002 000156' CMPB 2(SP),INDEX
729 001456 001360 BNE 10$
730 001460 022626 CMP (SP)+,(SP)+
731 001462 005067 000156' CLR INDEX
732 001466 005067 000160' CLR FLAGS
733 001472 RETURN
734
735
736
737
738 001474 012767 000200 000160' MPTAB: MOV #200,FLAGS
739 001502 112546 LFIL: MOVB (R5)+,(SP)
740 001504 112546 MOVB (R5)+,(SP)
741 001506 112566 000001 MOVB (R5)+,1(SP)
742 001512 010546 MOV R5,-(SP)
743 001514 011605 10$: MOV (SP),R5
744 001516 010446 20$: MOV R4,-(SP)
745 001520 005003 CLR R3
746 001522 152503 BISB (R5)+,R3
747 001524 022703 000000G CMPB #.EOF,R3
748 001530 001404 BEQ 30$
749 001532 106303 ASLB R3
750 001534 CALL @TABLE-2(R3)
751 001540 000767 BR 20$
752 001542 016700 000000G 30$: MOV $ADDR,R0
753 001546 042700 160000 BIC #160000,R0
754 001552 050200 BIS R2,R0
755 001554 066600 000004 ADD 4(SP),R0
756 001560 016003 000002 MOV 2(R0),R3
757 001564 046703 000154' BIC PRAPR,R3
758 001570 050203 BIS R2,R3
759 001572 011613 MOV (SP),(R3)
760 001574 040213 BIC R2,(R3)
761 001576 056713 000154' BIS PRAPR,(R3)
762 001602 012603 MOV (SP)+,R3
763 001604 005013 CLR (R3)
764 001606 040203 BIC R2,R3
765 001610 056703 000154' BIS PRAPR,R3
766 001614 010360 000002 MOV R3,2(R0)
767 001620 005267 000156' INC INDEX
768 001624 126667 000004 000156' CMPB 4(SP),INDEX
769 001632 001330 BNE 10$

```

```

: INDICATE MUX EXPANSION
: GET REPEAT COUNT
: SAVE CURRENT BINARY FILE POINTER
: RESTORE BINARY FILE POINTER
: GET FUNCTION CODE
: IS IT END OF FILE?
: IF EQ, YES
: CONVERT TO A WORD INDEX
: EXPAND A FIELD
: KEEP LOOKING FOR EOF
: TALLY ONE PASS THRU SECONDARY FILE
: ARE WE DONE?
: IF NE, NO
: PURGE STACK
: CLEAR SECONDARY INDEX
: AND SPECIAL FLAGS

```

```

: INDICATE MTP EXPANSION
: SAVE REPEAT COUNT
: AND LISTHEAD DISPLACEMENT
: SAVE BINARY POINTER
: RESET BINARY POINTER
: SAVE SECONDARY TABLE ADDRESS
: GET FUNCTION CODE
: IS IT END OF FILE?
: IF EQ, YES
: CONVERT TO WORD INDEX
: EXPAND A FIELD
: KEEP LOOKING FOR EOF
: GET MULTIPOINT LISTHEAD ADDRESS
: CLEAR CURRENT APR
: USE OUR APR
: ADD IN LIST HEAD DISPLACEMENT
: POINT AT LAST ENTRY IN LINKED LIST
: CLEAR OUT THE USER'S APR VALUE
: SET UP OUR OWN APR
: PLACE LINK TO NEW LAST ENTRY IN
: OLD LAST ENTRY, REPLACE OUR APR
: WITH USER'S APR
: POINT AT NEW LAST ENTRY
: ZERG OUT IT'S LINK
: REPLACE OUR APR VALUE WITH
: USER'S APR VALUE
: RESET POINTER TO LAST ENTRY
: TALLY ONE PASS THRU SECONDARY FILE
: ARE WE DONE?
: IF NE, NO

```


MACRO CROSS REFERENCE

CREF 04.00

MACRO NAME REFERENCES

ASL\$	#5-56	9-560								
CALL	7-273	7-276	7-289	12-725	12-750	14-829				
CALLR	14-842									
CRASH	8-399									
DHBD\$	#5-56	5-59								
RESRG	#5-56	8-372								
RETURN	7-291	8-303	8-305	8-315	8-317	8-325	8-334	8-340	8-348	8-353
	8-375	8-378	8-394	8-405	8-415	8-421	8-436	8-438	8-445	8-451
	8-462	8-475	8-486	8-488	8-501	8-503	8-511	8-513	8-521	8-523
	8-530	9-566	10-593	11-704	11-711	12-733	12-773	13-807	14-853	14-863
	14-865	14-872	14-878	15-886	15-891	15-896	15-901	15-906	15-911	15-916
	15-921	15-926	15-931	15-936	16-948	16-958	16-968	16-978	16-988	
SAVRG	#5-56	8-369								
SLTDF\$	#5-56	5-58								
SOB	9-562	10-582								
SWSTK\$	7-273									
TTAB\$	#5-76	6-94	6-95	6-96	6-97	6-98	6-99	6-100	6-101	6-102
	6-103	6-104	6-105	6-106	6-107	6-108	6-109	6-110	6-111	6-112
	6-113	6-114	6-115	6-116	6-117	6-118	6-119	6-120	6-121	6-122
	6-123	6-124	6-125	6-126	6-127	6-128	6-129	6-130	6-131	6-132
	6-133	6-134	6-135	6-136	6-137	6-138	6-139	6-140	6-141	6-142
	6-143	6-144	6-145	6-146						

508 000774

RETURN

SYMBOL	VALUE	REFERENCES							
LN.REF	= 000002	#5-60							
LN.SER	= 000002	#5-60							
LN.STA	= 000017	#5-60							
LN.SUB	= 000360	#5-60							
LN.TRI	= 000006	#5-60							
LSTHD	000614 R	5-106	#7-420						
LTAB	000620 R	5-107	#7-426						
L.COST	000015	#5-60							
L.CTL	000012	#5-60							
L.CVA	177776	#5-60	*7-358						
L.DDM	000002	#5-60	6-241						
L.DDS	000004	#5-60	8-530	9-599					
L.DLC	000003	#5-60							
L.DLM	000006	#5-60							
L.DLS	000010	#5-60							
L.FLG	000000	#5-60							
L.KRBA	000016	#5-60							
L.LEN	= 000022	#5-60							
L.MPF	000022	#5-60							
L.NMST	000020	#5-60							
L.NSTA	000014	#5-60							
L.OWNR	000021	#5-60							
L.UNT	000013	#5-60							
		12-812	7-354	7-371	7-434	7-454	8-519	9-588	12-786
		5-108	12-825	12-838					12-799
MPTAB	001110 R	5-108	#8-557						
MXPTB	001174 R	5-126	#9-580						
MXTAB	000776 R	5-109	#8-513						
M\$MGE	= 000000	4-4							
NOD	000602 R	5-110	#7-407						
NLPT	= ***** GX	10-668							
PECHA	000524 R	5-127	#7-369						
PRDCHA	000214 R	#5-149	*6-211	7-360	7-361				
PRFLAG	000346 R	#5-162	*6-203	7-456					
PR1	000624 R	5-111	#7-432						
PRPCHA	000224 R	#5-152	*6-212	7-373					
PRPRI	000222 R	#5-151	*6-213	7-437					
PRSCSR	000154 R	#5-148	*6-206	7-462					
PRSLTA	000220 R	#5-150							
PRTOV	000606 R	5-112	#7-413						
PR2HLD	000326 R	#5-157	*6-236	12-841					
PR2MBS	000246 R	#5-154	*6-221	12-802					
PR2MRT	000266 R	#5-155	*6-226	12-815					
PR2MWS	000226 R	#5-153	*6-216	12-789					
PR2RET	000306 R	#5-156	*6-231	12-828					
PR7	= ***** GX	7-436							
R\$MPL	= *****	5-62							
R\$11D	= *****	6-269	8-522	9-591					
R\$11M	= 000000	6-248							
SCOM	000656 R	5-113	#7-444						
SECSR	000666 R	5-114	#7-451						
SF.ACT	= 000200	#5-60							
SF.ENA	= 000100	#5-60							

TMPINI CREATED BY MACRO ON 29-JUN-85 AT 00:44 PAGE 2 M 8
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
L.NMST	000020	#5-59
L.NSTA	000014	#5-59
L.OWNR	000021	#5-59
L.UNT	000013	#5-59
M\$MGE	= 000000	7-202
RTSPC	000242 R	5-88 #6-176
R\$MPL	= *****	5-61 6-144 6-157 7-203
SF.ACT	= 000200	#5-59
SF.ENA	= 000100	#5-59
SF.LPB	= 000004	#5-59
SF.MFL	= 000040	#5-59
SF.PAC	= 000020	#5-59
SF.REA	= 000010	#5-59
SF.SER	= 000001	#5-59
SF.SVC	= 000002	#5-59
SF.UNL	= 000040	#5-59
S\$BAS	= *****	5-87 5-88 5-88
S.COST	000001	#5-59
S.FLG	000000	#5-59
S.LEN	000004	#5-59
S.NMST	000002	#5-59
S.OWNR	000003	#5-59
TMPEXT	000000 R	#5-80 6-171
ZF.PSE	= ***** GX	6-133
Z.DAT	= ***** GX	*6-184
Z.FLG	= ***** GX	6-133
\$ADDR	= ***** GX	*6-121 *6-138 *7-209 *7-222
\$AZFS	= ***** GX	6-172
\$BIN	= ***** GX	*6-118
\$CLOS	= ***** GX	6-174
\$CLOUE	= ***** GX	6-173
\$DFN	= ***** GX	6-131
\$ERRWZ	000042 R	#5-88 6-168
\$ERR8	000004 R	#5-87 6-185
\$FLAGS	= ***** GX	6-135 *6-141 6-181
\$FLAG1	= ***** GX	*6-126
\$FNOUT	= ***** GX	5-87
\$ICBLN	= ***** GX	7-210
\$LEFT	= ***** GX	*6-120
\$MXADD	= ***** GX	*6-122
\$MXSIZ	= ***** GX	*6-123
\$NALL	= ***** GX	*6-128
\$NAME	= ***** GX	6-130
\$NVECT	= ***** GX	*6-127
\$OFF	= ***** GX	*6-124
\$PDVA	= ***** GX	6-132 6-183
\$PRV.N	= ***** GX	5-87
\$PTR	= ***** GX	*6-119
\$SIZE	= ***** GX	*6-125
\$SLTA	= ***** GX	6-137
\$SZPTR	= ***** GX	6-129 *6-129
\$TM.IN	000000 RG	#6-117

```

250                                     .SBTTL .CSR,.END,.FILE
251                                     ;
252                                     ; ".CSR"
253                                     ;
254                                     STATES$ CSR
255                                     TRANS$ !MODIFY,NONE,$CSR
256
257                                     ;
258                                     ; ".END"
259                                     ;
260                                     STATES$ EOF
261                                     TRANS$ !POSEXP,EOF1,$END1
262                                     TRANS$ <','>,EOF2
263                                     TRANS$ !END,$EXIT
264                                     STATES$ EOF1
265                                     TRANS$ !END,$EXIT
266                                     TRANS$ <','>
267                                     STATES$ EOF2
268                                     TRANS$ !POSEXP,ONLY2,$END2
269                                     TRANS$ !ERROR,$EXIT
270
271                                     ;
272                                     ; ".FILE"
273                                     ;
274                                     STATES$ FILE
275                                     TRANS$ !FNAME,FILE1
276                                     TRANS$ !ERROR,$EXIT
277                                     STATES$ FILE1
278                                     TRANS$ !END,$EXIT,$FILE
279                                     TRANS$ <','>
280                                     STATES$
281                                     TRANS$ !POSEXP,$EXIT,$CFILE
282                                     TRANS$ !ERROR,$EXIT
283

```

```

55      ;***
56      ; LIBRARY MACROS
57      ;***
58      .MCALL ASR$,EMSG$,EMSG$R,NTLERS$,SLTDF$,CALLR
59      .IF DF R$$MPL
60
61      .MCALL KRBDF$
62
63      KRBDF$                ; DEFINE KRB OFFSETS
64
65      .ENDC
66
67      SLTDF$                ; DEFINE SLT OFFSETS
68
69      ;***
70      ; LOCAL DATA
71      ;***
72
73      .PSECT DATA,D
74
75      ;
76      ; NONSENSE INTERRUPT TABLE
77
78      NSITAB: .WORD NS0
79              .WORD NS1
80              .WORD NS2
81              .WORD NS3
82              .WORD NS4
83              .WORD NS5
84              .WORD NS6
85              .WORD NS7
86
87
88      K6TMP: .BLKW 1          ; $K6TAB TEMPORARY LOCATION
89      NSITMP: .BLKW 1        ; $NSX TEMPORARY LOCATION
90      FMKTMP: .BLKW 1        ; $FMASK TEMPORARY LOCATION
91
92      ;
93      ; ERROR MESSAGES
94      ;
95      .ENABL LC
96
97      NTLERS$ ,38,1,$TERR,$REP.P,,<Vector Setup Error>
98      .IF DF R$$MPL
99      NTLERS$ ,9D,1,$TERR,$REP.P,,<Unibus run not accessible>
100      .ENDC
101
102      .DSABL LC
103
104      ;
105      ; ERROR MESSAGE FORMAT STRING
106      ;
107      .ENABL LC
108      .NLIST BEX,,
109      .ASCIZ '*..'
110      .EVEN
111      .DSABL LC
112      .LIST BEX

```

TMPVCT CREATED BY MACRO ON 29-JUN-85 AT 00:47 PAGE 3 M 11

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
\$MXADD	= ***** GX	7-372 7-374
\$MXLEN	= ***** GX	7-394
\$NVECT	= ***** GX	6-141
\$OFF	= ***** GX	7-362
\$PDVA	= ***** GX	6-132 7-356 7-367
\$REP.P	= ***** GX	5-96
\$SLTA	= ***** GX	7-343
\$TERR	= ***** GX	5-96
\$TMLTA	000250 RG	#7-339
\$TMRDY	000564 RG	7-406 #8-419
\$TMVCT	000000 RG	#6-129
\$VECT1	= ***** GX	6-140
\$SVECT	= ***** GX	6-139

UNLPSI - NTL UNLOAD ROUTINES FD MACRO V05.03b Wednesday 17-Jul-85^{N 12} 12:05
Table of contents

5-	52	MACRO DEFINITIONS
6-	83	LOCAL SYMBOL DEFINITIONS
7-	95	\$XPSI - UNLOAD PSI DATA BASES
8-	140	DEADST - DEALLOCATE DESTINATION DESCRIPTORS
9-	184	XLDTE - DEALLOCATE LOCAL DTE BLOCK
10-	249	XPORT - DEALLOCATE PORT TABLE AND CIRCUIT BLOCKS
11-	295	XX29 - DEALLOCATE X.29 DATA BLOCK
12-	320	XTRA - DEALLOCATE TRACE DATA STRUCTURES

UNLPS1 CREATED BY MACRO ON 17-JUL-85 AT 12:05 PAGE 1 M 13
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
C\$LEN	000016	7-114
DEADST	000150 R	7-126
DEALST	= ***** GX	7-115 7-128 #8-154 7-124
D\$DATL	000030	7-118 7-121
D\$DTL	000031	8-168
D\$FLEN	000032	8-172
G\$LEN	000016	8-166
H\$CUG	000010	7-117
H\$DST	000012	7-116 7-119
H\$D29	000014	7-125 7-127
H\$LDTE	000002	7-127
H\$LEN	000042	9-205 *9-210
H\$NPT	000022	7-134
H\$PTB	000020	10-265
H\$PVC	000006	10-264
H\$RDTE	000004	7-113 7-116
H\$SVC	000036	7-119 7-122
H\$TRB	000016	7-122 7-125
H\$X29C	000040	12-339 *12-341
I\$AS	= *****	11-311 *11-313
KSAR5	= ***** GX	7-109 8-153 9-198 9-225 10-262 11-308 12-333 12-350
K\$LEN	000006	8-155 9-202 9-244 10-268 10-279 10-283 10-290
L\$CHLS	000034	9-218
L\$CHTB	000030	9-215
L\$CTEN	000032	9-222 *9-217
L\$LEN	000122	9-224
PSIPT	= ***** GX	9-213 9-228
Q\$LEN	000016	7-111 7-133 7-136 9-204 10-263 11-310 12-338
R\$LEN	000024	11-314
R\$R0	= 000002	7-120
R\$R1	= 000004	10-286
R\$R2	= 000006	#6-88
R\$R3	= 000010	#6-89
R\$R4	= 000012	#6-90
R\$R5	= 000014	#6-91
R\$RIS	= *****	#6-92 10-287
R\$R1D	= *****	#6-93
S\$LEN	000032	9-225 12-350
T\$CBUF	000004	9-225 12-350
T\$CLEN	000020	7-123
T\$CSIZ	000010	12-353
T\$CTCB	000016	12-347
T\$ST2	000034	12-349
T2\$STP	= 000020	12-342
XLDTL	000250 R	*12-344 12-344
XPORT	000436 R	7-130 #9-200
XTRA	000610 R	7-129 #10-263
XX29	000560 R	7-132 #12-335
X\$LEN	000106	7-131 #11-310
\$CEACX	= ***** GX	10-278
\$DEA16	= ***** GX	8-162 9-208 10-270
\$DECEX	= ***** GX	7-135 11-315
		9-219 9-235 12-362

UNLSUB - NTL UNLOAD SUBROUTINES MACRO V05.03b Saturday 29-Jun-85 ^{N 14} 00:48 Page 11-1
FNDADD - FIND LINE TABLE ADDRESS

278

UNLSUB - NTL UNLOAD SUBROUTINES MACRO V05.03b Saturday 29-Jun-85 ^{N 14} 00:48 Page 12
SCAN - SCAN SET FOR ACTIVITY

XLBR CREATED BY MACRO ON 29-JUN-85 AT 00:49 PAGE 2 M 15
MACRO CROSS REFERENCE CREF 04.00
MACRO NAME REFERENCES

CALL	6-97	6-99	6-137	6-141
RETURN	6-132	6-142		
SWS*KS	6-97			

XXBUF CREATED BY MACRO ON 29-JUN-85 AT 00:49 PAGE 1 M 16
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
XXBUF	000004 R	5-62 #5-65 5-70 6-92 6-97
\$BFLEN	= 001130 G	#5-58 5-63 5-65 6-97
\$XXAVL	001136 RG	#5-70 6-93 6-94 *6-96
\$XXBUF	000000 RG	#5-62 *6-95
\$XXINI	000000 RG	#6-91
\$XXLEN	000002 RG	#5-63 *6-99

```

112 000052      TTAB$  PRTOV,POP1
113 000054      TTAB$  SCOM,XSCOM
114 000056      TTAB$  SECSR,NULL
115 000060      TTAB$  SLNB,NULL
116 000062      TTAB$  SLNW,NULL
117 000064      TTAB$  STNB,NULL
118 000066      TTAB$  STNW,NULL
119 000070      TTAB$  UNB,NULL
120 000072      TTAB$  UNW,NULL
121 000074      TTAB$  WORD,POP2
122 000076      TTAB$  /R,POP1
123 000100      TTAB$  DPRB,POP1
124 000102      TTAB$  DPRW,POP2
125 000104      TTAB$  TIME,POP2
126 000106      TTAB$  MXPTB,XXMPTB
127 000110      TTAB$  PECHA,NULL
128 000112      TTAB$  X3CH01,NULL
129 000114      TTAB$  X3CH02,NULL
130 000116      TTAB$  X3CH03,NULL
131 000120      TTAB$  X3CH04,NULL
132 000122      TTAB$  X3CH05,NULL
133 000124      TTAB$  X3CH06,NULL
134 000126      TTAB$  X3CH07,NULL
135 000130      TTAB$  X3CH08,NULL
136 000132      TTAB$  X3CH09,NULL
137 000134      TTAB$  X3CH10,NULL
138 000136      TTAB$  X3CH11,NULL
139 000140      TTAB$  X2CH01,NULL
140 000142      TTAB$  X2CH02,NULL
141 000144      TTAB$  X2CH03,NULL
142 000146      TTAB$  X2CH04,NULL
143 000150      TTAB$  X2CH05,NULL
144
145      ;
146      ; SECONDARY FILE INDEX VALUE
147      ;
148 000152      INDEX: .BLKW 1
149
150      ;
151      ; MUX/MTP FLAGS
152      ;
153 000154      FLAGS: .BLKW 1 ; ZERU = NORMAL, NON-ZERO = MUX
154
155      ;
156      ; DATA NEEDED FOR SEARCH OF UNLOAD BLOCKS
157      ;
158 000156      UBIAS: .BLKW 1 ; BIAS/ADDRESS OF LIBRARY DESCRIPTOR
159 000160      UBA: .BLKW 1 ; OFFSET TO LIBRARY DESCRIPTOR
160 000162      BLOCK: .BLKW 32. ; LOCAL COPY OF UNLOAD BLOCK (FROM POOL..)
161
162 000000      .PSECT

```

```

714          .SBTTL X2CH
715          ;
716          ; MAXIMUM WINDOW SIZE
717          ;
718 001370 005767 000154' X2CH01: TST     FLAGS      ; MUX EXPANSION?
719 001374 001411          BEQ     10$      ; BR IF NO
720 001376 016700          MOV     INDEX,R0  ; GET SECONDARY FILE COUNTER
721 001402 126400 000013  CMPB    L,UNT(R4),R0 ; DO UNIT NUMBERS MATCH ?
722 001406 001004          BNE     10$      ; NO
723 001410 006300          ASL     R0        ; CONVERT TO A WORD INDEX
724 001412 116023 000000G MOV     $X2MWS(R0),(R3)+ ; STORE MAXIMUM WINDOW SIZE
725 001416 000401          BR      20$      ; AND EXIT
726 001420 005203          10$: INC     R3
727 001422          20$: RETURN
728          ;
729          ; MAXIMUM BLOCK SIZE
730          ;
731 001424 005767 000154' X2CH02: TST     FLAGS      ; MUX EXPANSION?
732 001430 001411          BEQ     10$      ; BR IF NO
733 001432 016700          MOV     INDEX,R0  ; GET SECONDARY FILE COUNTER
734 001436 126400 000013  CMPB    L,UNT(R4),R0 ; DO UNIT NUMBERS MATCH ?
735 001442 001004          BNE     10$      ; NO
736 001444 006300          ASL     R0        ; CONVERT TO A WORD INDEX
737 001446 016023 000000G MOV     $X2MBS(R0),(R3)+ ; STORE MAXIMUM BLOCK SIZE
738 001452 000401          BR      20$      ; AND EXIT
739 001454 005723          10$: TST     (R3)+
740 001456          20$: RETURN
741          ;
742          ; MAXIMUM RETRANSMIT COUNT
743          ;
744 001460 005767 000154' X2CH03: TST     FLAGS      ; MUX EXPANSION?
745 001464 001411          BEQ     10$      ; BR IF NO
746 001466 016700          MOV     INDEX,R0  ; GET SECONDARY FILE COUNTER
747 001472 126400 000013  CMPB    L,UNT(R4),R0 ; DO UNIT NUMBERS MATCH ?
748 001476 001004          BNE     10$      ; NO
749 001500 006300          ASL     R0        ; CONVERT TO A WORD INDEX
750 001502 116023 000000G MOV     $X2MRT(R0),(R3)+ ; STORE MAXIMUM RETRANSMIT COUNT
751 001506 000401          BR      20$      ; AND EXIT
752 001510 005203          10$: INC     R3
753 001512          20$: RETURN
754          ;
755          ; RETRANSMIT TIMER
756          ;
757 001514 005767 000154' X2CH04: TST     FLAGS      ; MUX EXPANSION?
758 001520 001411          BEQ     10$      ; BR IF NO
759 001522 016700          MOV     INDEX,R0  ; GET SECONDARY FILE COUNTER
760 001526 126400 000013  CMPB    L,UNT(R4),R0 ; DO UNIT NUMBERS MATCH ?
761 001532 001004          BNE     10$      ; NO
762 001534 006300          ASL     R0        ; CONVERT TO A WORD INDEX
763 001536 016023 000000G MOV     $X2RET(R0),(R3)+ ; STORE RETRANSMIT TIMER
764 001542 000401          BR      20$      ; AND EXIT
765 001544 005723          10$: TST     (R3)+
766 001546          20$: RETURN
767          ;
768          ; HOLDBACK TIMER
769          ;
770 001550 005767 000154' X2CH05: TST     FLAGS      ; MUX EXPANSION?

```

194 000206	PRINC: .BLKW 1	: INCREMENT SIZE IN 32. WORK BLOCKS
195 000210	PRPAR: .BLKW 2	: RAD50 PARTITION NAME
196 000214	PRDEV: .BLKW 1	: ASCII DEVICE NAME
197 000216	PRTBL: .BLKW 1	: DEVICE DISPATCH TABLE ADDRESS
198 000220	PRLINX: .BLKW 1	: LOGICAL LINK COUNT
199 000222	PRCTIM: .BLKW 1	: COUNTER TIMER VALUE
200 000224	PR3DBS: .BLKW 1	: DEFAULT BLOCK SIZE
201 000226	PR3MBS: .BLKW 1	: MAXIMUM BLOCK SIZE
202 000230	PR3DWS: .BLKW 1	: DEFAULT WINDOW SIZE
203 000232	PR3MWS: .BLKW 1	: MAXIMUM WINDOW SIZE
204 000234	PR3CAL: .BLKW 1	: CALL TIMER
205 000236	PR3CLR: .BLKW 1	: CLEAR TIMER
206 000240	PR3RES: .BLKW 1	: RESET TIMER
207 000242	PR3RST: .BLKW 1	: RESTART TIMER
208 000244	PR3MCL: .BLKW 1	: MAXIMUM CLEAR VALUE
209 000246	PR3MRE: .BLKW 1	: MAXIMUM RESETS VALUE
210 000250	PR3MRS: .BLKW 1	: MAXIMUM RESTARTS VALUE
211	:	
212	: LINE PARAMETERS	
213	:	
214 000252	PRSLTA: .BLKW 1	: SLT ADDRESS
215 000254	PRDLPV: .BLKW 1	: DLC PDV ADDRESS
216 000256	PRDDPV: .BLKW 1	: DDM PDV ADDRESS
217 000260	PRSLN: .BLKW 1	: SYSTEM LINE NUMBER
218 000262	PRCSR: .BLKB 48.	: MUX SLN'S
219 000342	PRPRI: .BLK 1	: CSR ADDRESS
220 000344	PRVECT: .BLKW 1	: INTERRUPT PRIORITY
221 000346	PRMTP: .BLKW 1	: VECTOR ADDRESS
222 000350	PRSNM: .BLKB 48.	: MULTIPOINT STATION COUNT
223 000352	PAPR: .BLKB 48.	: MULTIPOINT STATION NUMBERS
224 000432	PRDPR: .BLKW 1	: MULTIPOINT STATION ACTIVE POLLING RATIO
225 000512	PRMUX: .BLKW 1	: DEAD POLLING RATIO
226 000514	PRSCSR: .BLKW 16.	: MUX UNIT COUNT
227 000516	PRDCHA: .BLKW 2	: SECONDARY CSR TABLE
228 000556	PRPCHA: .BLKW 48.*2	: DEVICE CHARACTERISTICS
229 000562	:	: MUX CHARACTERISTICS
230 001062	:	: PROTOCOL EMULATOR CHARACTERISTICS
231 001064	:	: MUX CHARACTERISTICS
232 001224	PRLSAD: .BLKW 1	: LOCAL PHYSICAL STATION ADDRESS
233 001226	PR2MWS: .BLKW 8.	: MAXIMUM WINDOW SIZE
234 001246	PR2MBS: .BLKW 8.	: MAXIMUM BLOCK SIZE
235 001266	PR2MRT: .BLKW 8.	: MAXIMUM RETRANSMIT COUNT
236 001306	PR2RET: .BLKW 8.	: RETRANSMIT TIMER
237 001326	PR2HLD: .BLKW 8.	: HOLDBACK TIMER
238		
239 000000	.PSECT	

770 001634 062706 000006
771 001640 005067 000156'
772 001644 005067 000160'
773 001650

ADD #6,SP
CLR INDEX
CLR FLAGS
RETURN

; PURGE STACK
; CLEAR SECONDARY INDEX
; AND SPECIAL FIAGS


```

510 ;
511 ; SECONDARY FILE
512 ;
513 000776 005267 000352' MXTAB: INC FLAGS ; INDICATE MUX EXPANSION
514 001002 112546 FILE: MOV (R5)+,-(SP) ; GET REPEAT COUNT
515 001004 010546 MOV R5, -(SP) ; SAVE CURRENT BINARY POINTER
516 001006 011605 10$: MOV (SP), R5 ; RESTORE BINARY POINTER
517 001010 005767 000352' TST FLAGS ; IS IT MUX EXPANSION ?
518 001014 001407 BEQ 20$ ; NO ..
519 001016 126467 000013 000350' CMPB L, UNT(R4), INDEX ; DO UNIT NUMBERS MATCH ?
520 001024 001003 BNE 20$ ; NO ..
521 ;
522 .IF DF R$11D ! 1$AS
523 MOV R3, -(SP) ; GET LINE TABLE ADDRESS
524 BIC #160000, (SP) ; CLEAR MAPPING BITS
525 BIS #60000, (SP) ; SLT LINE TABLE ADDRESS USES APR3
526 CMP L, DDS(R4), (SP)+ ; IS THE LINE TABLE POINTER CORRECT ?
527 ;
528 .IFF
529 ;
530 001026 026403 000004 CMP L, DDS(R4), R3 ; IS THE LINE TABLE POINTER CORRECT?
531 .ENDC
532 ;
533 001032 001025 BNE 101$ ; NO .. ERROR
534 001034 005002 20$: CLR R2 ; GET FUNCTION CODE
535 001036 152502 BISB (R5)+, R2 ;
536 001040 122702 000000G CMPB #, EOF, R2 ;
537 001044 001404 BEQ 30$ ; IS IT END OF FILE?
538 001046 106302 ASLB R2 ; IF EQ, YES
539 001050 CALL @TABLE-2(R2) ; CONVERT TO A WORD INDEX
540 001054 000767 BR 20$ ; EXPAND A FIELD
541 001056 005267 000350' 30$: INC INDEX ; KEEP LOOKING FOR EOF
542 001062 126667 000002 000350' CMPB 2(SP), INDEX ; TALLY ONE PASS THRU SECONDARY FILE
543 001070 001346 BNE 10$ ; ARE WE DONE?
544 001072 022626 CMP (SP)+, (SP)+ ; IF NE, NO
545 001074 005067 000350' CLR INDEX ; PURGE STACK
546 001100 005067 000352' CLR FLAGS ; CLEAR SECONDARY INDEX
547 001104 RETURN ; AND SPECIAL FLAGS
548 ;
549 ;
550 ; ERROR CONDITION
551 ;
552 001106 101$: CRASH ; LINE TABLE POINTER (R3) IS SCREWED UP
553 ;
554 ;
555 ; LINKED SECONDARY FILE
556 ;
557 001110 MPTAB: CRASH
558 001112 112546 LFILE: MOV (R5)+, -(SP) ; SAVE REPEAT COUNT
559 001114 122525 CMPB (R5)+, (R5)+ ; DISCARD LISTHEAD DISPLACEMENT
560 001116 010546 MOV R5, -(SP) ; SAVE CURRENT BINARY POINTER
561 001120 011605 10$: MOV (SP), R5 ; RESET BINARY POINTER
562 001122 005002 20$: CLR R2 ; GET FUNCTION CODE
563 001124 152502 BISB (R5)+, R2 ;
564 001126 022702 000000G CMPB #, EOF, R2 ; IS IT END OF FILE?
565 001132 001404 BEQ 30$ ; IF EQ, YES
566 001134 106302 ASLB R2 ; CONVERT TO WORD INDEX

```

SYMBOL CROSS REFERENCE

CREF 04.00

SYMBOL VALUE REFERENCES

SF.LPB	= 000004	#5-60				
SF.MFL	= 000040	#5-60				
SF.PAC	= 000020	#5-60				
SF.REA	= 000010	#5-60				
SF.SER	= 000001	#5-60				
SF.SVC	= 000002	#5-60				
SF.UNL	= 000040	#5-60				
SLNB	000740 R	5-115	#7-470			
SLNW	000744 R	5-116	#7-476			
STNB	000750 R	5-117	#7-482			
STNW	000754 R	5-118	#7-488			
S.COST	000001	#5-60				
S.FLG	000000	#5-60				
S.LEN	000004	#5-60				
S.NMST	000002	#5-60				
S.OWNR	000003	#5-60				
TABLE	000002 R	#5-91	6-266	8-539	8-567	9-608
TEMP	000000 R	#5-86	+9-627	+9-628	9-629	*9-630
TIME	001314 R	5-125	#9-627			9-631
UBA	000356 R	#5-178	6-245	6-255	*10-667	*10-710
UBIAS	000354 R	#5-177	*10-711			
UNB	000760 R	5-119	#7-494			
UNW	000764 R	5-120	#7-500			
WORD	000770 R	5-121	#7-506			
X2CH01	001644 R	5-139	#12-783			
X2CH02	001700 R	5-140	#12-796			
X2CH03	001734 R	5-141	#12-809			
X2CH04	001770 R	5-142	#12-822			
X2CH05	002024 R	5-143	#12-835			
X3CH01	001570 R	5-128	#11-726			
X3CH02	001574 R	5-129	#11-731			
X3CH03	001600 R	5-130	#11-736			
X3CH04	001604 R	5-131	#11-741			
X3CH05	001610 R	5-132	#11-746			
X3CH06	001614 R	5-133	#11-751			
X3CH07	001620 R	5-134	#11-756			
X3CH08	001624 R	5-135	#11-761			
X3CH09	001630 R	5-136	#11-766			
X3CH10	001634 R	5-137	#11-771			
X3CH11	001640 R	5-138	#11-776			
ZTIME	= ***** GX	9-627				
\$ADDR	= ***** GX	6-239				
\$BIAS	= ***** GX	6-252				
\$BIN	= ***** GX	6-254				
\$FLAGS	= ***** GX	6-203				
\$MAPX	= ***** GX	6-253	10-672			
\$PTR	= ***** GX	6-260				
\$SLTA	= ***** GX	6-240				
\$SLTSV	= ***** GX	7-357				
\$TMEX7	000000 RG	#6-200				
\$X2HLD	= ***** GX	6-236				
\$X2MBS	= ***** GX	6-221				

IMPIN: CREATED BY MACRC ON 29-JUN-85 AT 00:44 PAGE 3 N 8
SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
\$TWAR	= ***** GX	5-88
\$TWE	= ***** GX	5-87
\$XBUF2	= ***** GX	6-118 6-119
\$XLEN2	= ***** GX	6-120
\$CSR	= ***** GX	6-166
\$VECT	= ***** GX	6-164

```

285 .SBTTL .INT,.INT1,.INT2,.INT3,.LIBR,.LFILE
286
287 ;
288 ; ".INT", ".INT1", ".INT2", OR ".INT3"
289 ;
290 000002 STATES$ INT
291 000002 TRANS$ !SYMBOL,ONLY1,$INTX
292 000002 TRANS$ !ERROR,$EXIT
293
294 ;
295 ; ".LIBR"
296 ;
297 000002 STATES$ LIBR
298 000002 TRANS$ !FNAME1,LIBR1,$LIBR1
299 000002 TRANS$ !FNAME1,LIBR1,.ABF1
300 000002 TRANS$ !ERROR,$EXIT
301 000002 STATES$ LIBR1
302 000002 TRANS$ !END,$EXIT,$LIBR
303 000002 TRANS$ <','>
304 000002 STATES$
305 .IF DF M$$MGE
306 000002 TRANS$ !POSEXP,LIBR2,$CORE2
307 .IFF
308 TRANS$ !EAT,ONLY2,$LIBR
309 .IFTF
310 000002 TRANS$ !ERROR,$EXIT
311 .IFT
312 000002 STATES$ LIBR2
313 000002 TRANS$ $LAMDA,ONLY2,$LIBR
314 .ENDC
315
316 ;
317 ; ".LFILE"
318 ;
319 000002 STATES$ LFILE
320 000002 TRANS$ !FNAME,LFILE1
321 000002 TRANS$ !ERROR,$EXIT
322 000002 STATES$ LFILE1
323 000002 TRANS$ <','>
324 000002 STATES$
325 000002 TRANS$ !POSEXP,LFILE2
326 000002 TRANS$ !ERROR,$EXIT
327 000002 STATES$ LFILE2
328 000002 TRANS$ !ONLY2,$EXIT,$LFILE
  
```

112
113 000000

.PSECT

IMPVCT CREATED BY MACRO ON 29-JUN-85 AT 00:47 PAGE 4 N 11
MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
ASR\$	#5-58 6-159
CALL	6-151 6-315 7-406 8-431
CALLR	#5-58 6-147
CRASH	6-320 9-467 9-482
EMSG\$	#5-58 6-315
EMSG\$R	#5-58
MIPS	6-228 6-268
NTLR\$	#5-58 5-96
RETURN	6-274 6-316 7-417 8-447 9-498
SLTDF\$	#5-58 5-67
SOB	6-165 6-264 9-466
SWSTK\$	6-151

.TITLE UNLPS1 - NTL UNLOAD ROUTINES FOR PSI
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

NTL - UNLOAD ROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 2.00 06-AUG-81
DECNET-11M/S V3.1
DECNET-11M-PLUS V1.1
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

UNLPSI CREATED BY MACRO ON 17-JUL-85 AT 12:05 PAGE 2 N 13
SYMBOL CROSS REFERENCE CREF 04.00
SYMBOL VALUE REFERENCES
\$XDEAC = ***** GX 8-177 9-241 10-281 10-289
\$XPSI 000000 RG #7-111

```

280                                     .SBTTL  SCAN - SCAN SLT FOR ACTIVITY
281
282                                     ;+
283                                     SCAN - SCAN SLT FOR ACTIVITY
284
285                                     INPUTS:
286                                     R5 - SLT ADDRESS
287
288                                     OUTPUTS:
289                                     R0 - COUNT OF OTHER LINES USING THIS DLC
290                                     R1 - COUNT OF OTHER LINES USING THIS DDM
291                                     R2 - COUNT OF OTHER LINES ON THIS CONTROLLER
292
293                                     -
294
295 000136 005000 SCAN:: CLR R0 ; OTHER READY LINES USING THIS DLC
296 000140 005001 CLR R1 ; OTHER READY LINES USING THIS DDM
297 000142 005067 177632 CLR UNITS ; OTHER READY LINES ON THIS CONTROLLER
298 000146 017702 000000G MOV @SLTMA,R2 ; GET SLT TABLE ADDRESS
299
300 000152 017704 000000G MOV @SLTNM,R4 ; GET NUMBER OF SLT'S
301 000156 012203 10$: MOV (R2)+,R3 ; GET NEXT SLT
302
303 000160 032763 040000 000000 BIT #LF.RDY,L.FLG(R3) ; IS THIS LINE READY?
304 000166 001420 BEQ 30$ ; IF EQ, NO
305 000170 126365 000003 000003 CMPB L.DLC(R3),L.DLC(R5) ; SAME DLC AS OURS?
306 000176 001001 BNE 20$ ; IF NE, NO
307 000200 005200 INC R0 ; TALLY IT
308 000202 126365 000002 000002 20$: CMPB L.DDM(R3),L.DDM(R5) ; SAME DDM AS OURS?
309 000210 001007 BNE 30$ ; IF NE, NO
310 000212 005201 INC R1 ; TALLY IT
311 000214 126365 000012 000012 CMPB L.CTL(R3),L.CTL(R5) ; SAME CONTROLLER AS OURS?
312 000222 001002 BNE 30$ ; IF NE, NO
313 000224 005267 177550 INC UNITS ; TALLY IT
314 000230 30$: SOB R4,10$ ; LOOP <SLT COUNT> LIMES
315 000234 010067 000000G MOV R0,DLCAC ; SETUP ACTIVITY COUNTS
316 000240 010167 000000G MOV R1,DDMAC ;
317 000244 016702 177530 MOV UNITS,R2 ;
318 000250 RETURN ;
319
320 000001 .END

```

FILEID**xPAR

```

XX      XX  PPPPPPP  AAAAAA  RRRRRRRR
XX      XX  PPPPPPP  AAAAAA  RRRRRRRR
XX      XX  PP      PP  AA      AA  RR      RR
XX      XX  PP      PP  AA      AA  RR      RR
XX      XX  PP      PP  AA      AA  RR      RR
XX      XX  PP      PP  AA      AA  RR      RR
XX      XX  PPPPPPP  AA      AA  RRRRRRRR
XX      XX  PPPPPPP  AA      AA  RRRRRRRR
XX      XX  PP      AA      AA  RR      RR
XX      XX  PP      AAAAAAAA  RR      RR
XX      XX  PP      AAAAAAAA  RR      RR
XX      XX  PP      AA      AA  RR      RR
XX      XX  PP      AA      AA  RR      RR
XX      XX  PP      AA      AA  RR      RR
XX      XX  PP      AA      AA  RR      RR

```

```

....
....
....
....

```

```

LL      SSSSSSSS  TTTTTTTTTT
LL      SSSSSSSS  TTTTTTTTTT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SSSSSS  TT
LL      SSSSSS  TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LLLLLLLLLL  SSSSSSSS  TT
LLLLLLLLLL  SSSSSSSS  TT

```


XXBUF CREATED BY MACRO ON 29-JUN-85 AT 00:49 PAGE 2 **B 1**
MACRO CROSS REFERENCE CREF 04.00
MACRO NAME REFERENCES
RETURN 6-103

```

311 000442 110022          MOVB    R0,(R2)+      ; ...
312 000444 062767 000002 000000G    ADD     #2,$SIZE    ; ADJUST DATA BASE SIZE
313 000452          CALL     $RQ1                ; REQUEST BINARY FILE AGAIN
314 000456 103402          BCS     30$           ; IF CS, NO ACCESS
315 000460 112712 000000G    MOVB    #.EOF,(R2)   ; STORE END OF FILE FUNCTION CODE
316 000464          30$:    RETURN
317          ;
318          ; ERROR CONDITIONS
319          ;
320 000466          101$:    EMSG$R 22             ; FILE COUNT TOO LARGE
321 000474          111$:    EMSG$R KX             ; RESOURCE ALLOCATION FAILURE
322

```

MACRO CROSS REFERENCE

CREF 04.00

MACRO NAME REFERENCES

ASR\$	#5-58	18-542								
CALL	6-141	6-142	7-169	8-190	9-209	10-228	11-247	12-270	12-271	12-273
	12-280	12-281	12-287	12-288	12-305	12-313	14-366	14-368	14-378	14-388
	14-396	14-397	14-399	15-433	15-434	15-436	15-440	15-445	15-451	15-456
	15-457	15-459	16-482	16-483	16-486	17-511	18-529	18-544	19-572	19-580
	6-146	8-194	9-213	10-232	11-251					
CALLR	#5-58	17-511								
EMSG\$	#5-58	6-154	12-320	12-321	13-348	14-404	14-413	14-414	14-415	16-495
EMSG\$R	18-553	18-554	18-555							
	#5-58	18-544								
GETRV	#5-58	5-92	5-94	5-95	5-96	5-98	5-99	5-101	5-103	5-105
NTLR\$	5-106	5-107	5-108							
RAND	#18-544	18-544								
RETURN	6-149	7-175	12-316	13-344	14-408	15-442	15-467	16-490	17-512	18-548
	19-582									

```

56      ;**
57      ; LIBRARY MACROS
58      ;**
59      .MCALL ASR$,EMSG$,EMSG$R,NTLERS$,PUTRC,PUTAD,ASL$
60      .ENABL LC
61      .ENABL LC
62      .ENABL LC
63
64      ;**
65      ; LOCAL DATA
66      ;**
67
68 000000      .PSECT DATA,D
69
70      .NLIST BEX
71
72      ;
73      ; MAPPING DISPLACEMENTS
74      ;
75 000000 000000G      .WORD .BASEB      ; DEFAULT MAPPING ASSIGNMENT
76 000002 000000 020000 040000 MAPP: .WORD 0,20000,40000,60000,100000,120000,140000,160000
77
78      ;
79      ; MAPPING DISPLACEMENTS CONVERTED TO BYTE VALUES
80      ;
81 000022 0000C      .BYTE <<,3BASEB/2>&77777>/10000      ; DEFAULT MAPPING
82 000023 000 001 002 MAPBYT: .BYTE 0,1,2,3,4,5,6,7
83      .EVEN
84
85      ;
86      ; ERROR MESSAGES
87      ;
88 000034      NTLERS$ ,26,2,$TERR,$REP.C,$ELINE,<Resource allocation failure.>
89 000102      NTLERS$ ,39,2,$TWARN,$REP.C,$ELINE,<Invalid mapping register.>
90 000144      NTLERS$ ,2A,6,$TWARN,REP6,$ELINE,<Not built for APR*.>
91 000200      NTLERS$ ,76,6,$TWARN,$REP.C,$ELINE,<Base address not zero.>
92 000240      NTLERS$ ,77,6,$TERR,$CUR.N,$ELINE,<Read failure ( -***. ).>
93 000300      NTLERS$ ,78,2,$TERR,$REP.C,$ELINE,<Maximum # allocations/libraries is 14.>
94
95      ;
96      ; ERROR MESSAGE FORMAT STRINGS
97 000360 052 040 124 FMT2: .ASCIZ '* Template -- '
98 000377 052 040 111 FMT6: .ASCIZ '* Image file -- '
99      .EVEN
100
101 000000      .PSECT

```


VAC4 CREATED BY MACRO ON 29-JUN-85 AT 02:09 PAGE 1 B 5

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES							
ALLERR	= 000712 R	9-305	#9-335	10-364					
FLCHA	= ***** GX	6-124	9-293	10-353					
FMT2	= 000360 R	5-88	5-89	5-93	#5-97				
FMT6	= 000377 R	5-90	5-91	5-92	#5-98				
FM.2	= 000000	#5-88	#5-89	#5-93					
FM.6	= 000000	#5-90	#5-91	#5-92					
F1.UMR	= ***** GX	6-142	9-300						
F2.FIL	= ***** GX	6-121							
IS\$AS	= *****	7-211	13-449						
KISARC	= ***** GX	8-242	8-246						
MAPBYT	= 000023 R	#5-82	7-214	8-260					
MAPP	= 000002 R	#5-76	8-258						
MAX14	= 001046 R	6-139	9-298	10-358	#11-387				
REP6	= 001100 R	5-90	#12-413						
R\$EIS	= *****	7-211	13-449						
R\$SMPL	= *****	8-241							
R\$S11D	= *****	7-211	13-449						
S\$SBAS	= *****	5-88	5-88	5-89	5-89	5-90	5-90	5-91	5-92
		5-92	5-93	5-93					
UISARU	= ***** GX	8-246							
ZER16	= 001200 R	10-365	#13-456						
ZER19	= 001136 R	9-306	#13-445						
\$ALL16	= ***** GX	10-363							
\$ALL18	= ***** GX	6-146	9-304						
\$ALN18	= ***** GX	6-144	9-302						
\$BFALL	= ***** GX	6-141	6-161	6-162					
\$BFBAS	= ***** GX	6-130	6-135						
\$BFLBN	= ***** GX	6-174	6-175						
\$BFPTR	= ***** GX	6-149							
\$BFR	= ***** GX	13-462							
\$BFXFR	= ***** GX	6-176							
\$BINP	= 000004 RG	#6-120							
\$BING	= 000000 RG	#6-117							
\$CAPR	= ***** GX	*7-214	*8-260	12-421					
\$CLDEQ	= ***** GX	6-181	6-187						
\$CLIOS	= ***** GX	6-179							
\$CORE	= 000510 RG	#9-280							
\$CORE1	= 000312 RG	#7-201							
\$CORE2	= 000400 RG	6-118	#8-232	9-278					
\$CORK	= 000504 RG	#9-277							
\$CUR.N	= ***** GX	5-92							
\$CVAL	= ***** GX	*7-212	*7-217	*9-295	9-296	9-320	9-321		
\$ELINE	= ***** GX	5-88	5-89	5-90	5-91	5-92	5-93		
\$ERRZA	= 000144 R	#5-90	6-132						
\$ERR26	= 000034 R	#5-88	6-188	9-335					
\$ERR39	= 000102 R	#5-89	8-256						
\$ERR76	= 000200 R	#5-91	6-138						
\$ERR77	= 000240 R	#5-92	6-180						
\$ERR78	= 000300 R	#5-93	11-398						
\$FLAGS	= ***** GX	6-124	9-293	10-353					
\$FLAG1	= ***** GX	6-142	9-300						
\$FLAG2	= ***** GX	6-121							

VAC4 CREATED BY MACRO ON 29-JUN-85 AT 02:09 PAGE 2 C 5

FILEID**VAC6

```

VV      VV      AAAAAA      CCCCCCCC      666666
VV      VV      AAAAAA      CCCCCCCC      666666
VV      VV      AA      AA      CC      66
VV      VV      AA      AA      CC      66
VV      VV      AA      AA      CC      66
VV      VV      AA      AA      CC      66
VV      VV      AA      AA      CC      66666666
VV      VV      AA      AA      CC      66666666
VV      VV      AAAAAAAAAA      CC      66      66
VV      VV      AAAAAAAAAA      CC      66      66
VV      VV      AA      AA      CC      66      66
VV      VV      AA      AA      CC      66      66
VV      VV      AA      AA      CC      66      66
VV      VV      AA      AA      CCCCCCCC      666666
VV      VV      AA      AA      CCCCCCCC      666666

```

```

....
....
....
....

```

```

LL      SSSSSSSS      TTTTTTTTTT
LL      SSSSSSSS      TTTTTTTTTT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SSSSSS      TT
LL      SSSSSS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SSSSSSSS      TT
LLLLLLLLLLLL      SSSSSSSS      TT
LLLLLLLLLLLL      SSSSSSSS      TT

```

```

342                                     .SBTTL $X2CHB - .X2CHB ACTION ROUTINE
343                                     .SBTTL $X3CHB - .X3CHB ACTION ROUTINE
344
345                                     ;+
346                                     ;
347                                     ; $X2CHB - .X2CHB ACTION ROUTINE
348                                     ; $X3CHB - .X3CHB ACTION ROUTINE
349                                     ;
350                                     ; INPUTS:
351                                     ; $VALUE - VALUE OF FIRST OPERAND (INDICATES WHICH PARAMETER
352                                     ; FROM X2P$DF OR X3P$DF TO USE)
353                                     ;
354                                     ; OUTPUTS:
355                                     ; R1 DESTROYED
356                                     ;
357                                     ; -
358                                     .ENABL LSB
359 000700 012701 000000G $X2CHB::MOV #.X2CH1,R1 ; GET FUNCTION CODE
360 000704 026727 000000G 000001 CMP $VALUE,#X2BLO ; IS VALUE WITHIN RANGE?
361 000712 103425 BLO 10$ ; BR IF NO
362 000714 026727 000000G 000003 CMP $VALUE,#X2BHI ; WITHIN RANGE?
363 000722 101021 BHI 10$ ; BR IF NO
364 000724 026727 000000G 000002 CMP $VALUE,#X2BLK ; BLOCK SIZE PARAMETER?
365 000732 001415 BEQ 10$ ; YES, NOT ALLOWED AS BYTE
366 000734 CALL $VXLLC ; MAKE SURE IT'S NOT AN LLC
367 000740 000417 BR 30$
368
369 000742 012701 000000G $X3CHB::MOV #.X3C01,R1 ; GET FUNCTION CODE
370 000746 026727 000000G 000003 CMP $VALUE,#X3BLO ; IS OPERAND WITHIN RANGE?
371 000754 103404 BLO 10$ ; BR IF NO
372 000756 026727 000000G 000013 CMP $VALUE,#X3BHI ; WITHIN RANGE?
373 000764 101403 BLOS 20$ ; BR IF YES
374 000766 10$ EMSGS 91 ; INVALID OPERAND
375 000774 20$ CALL $VFLLC ; MAKE SURE IT IS AN LLC
376 001000 066701 000000G 30$ ADD $VALUE,R1 ; POINT TO CORRECT FUNCTION CODE
377 001004 005301 DEC R1 ; OPERAND STARTS AT 1
378 001006 CALL $RQ1 ; REQUEST BINARY BUFFER
379 001012 110122 MOVB R1,(R2)+ ; STORE FUNCTION CODE
380 001014 005267 000000G INC $SIZE ; ADJUST DATA BASE SIZE
381 001020 RETURN
382 .DSABL LSB

```

```

203      ;+
204      ; LINK - LINK NEWLY ALLOCATED LIBRARY BLOCK INTO LIST
205      ;
206      ; INPUTS:
207      ;     LBLK=LIBRARY BLOCK ADDRESS
208      ;
209      ; OUTPUTS:
210      ;     NONE
211      ; -
212      LINK:  MJV      $NTLHB+.CXLBR,R0 ; GET FIRST LIBRARY BLOCK BIAS
213      000340 016700 000000C          BNE      20$      ; IF NE, THERE IS AT LEAST ONE
214      000344 001005                  MOV      LBLK,$NTLHB+.CXLBR ; SET FIRST LIBRARY BLOCK BIAS
215      000346 016767 000000' 000000C BR       30$      ;
216      000354 000411                  10$:  MOV      (R1),R0      ; GET NEXT LIBRARY BLOCK BIAS
217      000356 011100                  20$:  CALL     $GETLB      ; READ IN A 32. WORD BLOCK
218      000364 005711                  TST      (R1)      ; IS THIS THE LAST BLOCK IN LIST?
219      000366 001373                  BNE      10$      ; IF NE, NO
220      00037C 016711 000000'          MOV      LBLK,(R1)    ; SET LAST LIBRARY BLOCK IN LIST BIAS
221      000374                  CALL     $PUTLB      ; RE-WRITE LIBRARY BLOCK
222      000400                  30$:  RETURN

```

Symbol table

A\$\$CHK= 000000	F\$\$LVL= 000001	L\$\$P11= 000001	N\$\$NCT= 000001	X\$\$DBT= 000000
A\$\$CPS= 000000	G\$\$TPP= 000000	L\$\$11R= 000000	N\$\$PEM= 000001	\$ALB= 000000RG
A\$\$PRI= 000000	G\$\$TSS= 000000	M\$\$CPB= 000174	P\$\$P45= 000000	\$ALN18= ***** GX
A\$\$TRP= 000000	G\$\$TTK= 000000	M\$\$CRX= 000000	P\$\$WRD= 000000	\$ELINE= ***** GX
C\$\$CKP= 000000	G\$\$WRD= 000000	M\$\$FCS= 000000	Q\$\$OPT= 000010	\$ERRY1= 000000R
C\$\$ORE= 000400	I\$\$RAR= 000000	M\$\$MGE= 000000	R\$\$DER= 000000	\$LBADR= ***** GX
C\$\$RSH= 177564	I\$\$RDN= 000000	M\$\$NET= 000000	R\$\$K11= 000001	\$LBERR= ***** GX
D\$\$BUG= 177514	K\$\$CNT= 177546	M\$\$OVR= 000000	R\$\$SND= 000000	\$LBIAS= ***** GX
D\$\$ISK= 000000	K\$\$CSR= 177546	N\$\$ACC= 000001	R\$\$11M= 000000	\$PUTLB= ***** GX
D\$\$L11= 000001	K\$\$LDC= 000000	N\$\$BUF= 000001	S\$\$WRG= 000000	\$REP.C= ***** GX
D\$\$YNC= 000000	K\$\$TPS= 000074	N\$\$LDV= 000001	S\$\$YSZ= 007600	\$TERR= ***** GX
D\$\$YNM= 000000	LBLK = ***** GX	N\$\$MCP= 000001	T\$\$KMG= 000000	.LBIAS= ***** GX
E\$\$XPR= 000000	LD\$LP = 000000	N\$\$MLL= 000001	T\$\$MIN= 000000	.LBUFF= ***** GX
FM2 = ***** GX	L\$\$ASG= 000000	N\$\$MOV= 000010	V\$\$CTR= 001000	.MGMGE= ***** GX
FM.2 = 000000	L\$\$DRV= 000000			

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
 000120 001 (RW,I,LCL,REL,CON)
 DATA 000052 002 (RW,D,LCL,REL,CON)
 Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 10154 Words (40 Pages)
 Size of core pool: 14440 Words (55 Pages)
 Operating system: RSX-11M/PLUS

Elapsed time: 00:00:06.36
 SY:VAC8.V2,[132,134]VAC8/CR/-SP=SY:[1,1]RSXMC.M.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMC.M/PA:1,[132,10]VAC8

```

LL          SSSSSSSS   TTTT TTTT TTTT
LL         SSSSSSSS   TTTT TTTT TTTT
           SS        TT
           SS        TT
           SS        TT
           SS        TT
           SSSSSS    TT
           SSSSSS    TT
                   SS  TT
                   SS  TT
                   SS  TT
                   SS  TT
LLLLLLLLLLL SSSSSSSS TT
LLLLLLLLLLL SSSSSSSS TT

```

```

510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527 000662
528 000666 005067 000000G
529 000672 116700 000000G
530 000676 116701 000000G
531 000702 060001
532 000704 001403
533 000706 006301
534 000710 006301
535 000712 005721
536 000714 022121
537 000716 005721
538 000720 010167 000000G
539
540 000724 016767 000000C 000000G
541 000732 001466
542 000734 012767 000100 000000G
543 000742 012702 000000G
544 000746 005077 000000G
545 000752 016777 000000G 000000G
546
547
548
549 000760
550 000766
551 000772 010200
552 000774 062700 000100
553 001000 010203
554 001002 005723
555 001004 020300
556 001006 001426
557 001010 005723
558 001012 005723
559 001014 021412
560 001016 032723 000001
561 001022 001770
562 001024 112304
563 001026 112305
564 001030 060405
565 001032 006305
566 001034 006305

;
*
UBA - CALCULATE UNLOAD BLOCK ADDRESS

INPUTS:
$ADDR=LINE TABLE ADDRESS
$SIZE=LINE TABLE SIZE
$NALL,$NLIB=NUMBER OF ALLOCATIONS/LIBRARIES
$NTLPT=POINTER TO NTL DATA BLOCK
DATA BLOCK+.$CXUNL=POINTER TO FIRST UNLOAD BLOCK

OUTPUTS:
$UBIAS=BIAS OF UNLOAD BLOCK (MAPPED SYSTEM ONLY)
$UDA=UNLOAD DATA ADDRESS FOR FIRST ALLOCATION ENTRY
$LDA=UNLOAD DATA ADDRESS FOR FIRST LIBRARY DESCRIPTOR ENTRY

-

UBA: CALL GETIND ; GET THE PDV INDEX
CLR $UDA ; SET NO UNLOAD DATA ADDRESS YET
MOVB $NALL,R0 ; GET NUMBER OF ALLOCATIONS
MOVB $NLIB,R1 ; AND NUMBER OF LIBRARIES
ADD R0,R1 ; ADD THEM TOGETHER
BEQ 10$ ; IF ZERO, NO ALLOCS/LIBRS
ASL R1 ; MULTIPLY BY 4 (2 WORDS PER UNLOAD ENTRY)
ASL R1
TST (R1)+ ; INCLUDE 1 WORD FOR ALLOC/LIBR COUNT
10$: CMP (R1)+,(R1)+ ; INCLUDE 2 WORDS FOR LINE TABLE ADDRESS/SIZE
TST (R1)+ ; INCLUDE 1 WORD FOR PDV INDEX AND RESERVED BYTE
MOV R1,$LDA ; SAVE UNLOAD SIZE NEEDED

MOV .CXUNL+$NTLHB,$UBIAS ; GET BIAS OF FIRST UNLOAD BLOCK
BEQ 70$ ; IF NONE, GO ALLOCATE ONE
MOV #64,$RSIZE ; SET SIZE OF NETWORK POOL BLOCKS
MOV #NNTLUB,R2 ; SET VIRTUAL ADDR. OF UNLOAD BLOCK BUFFER
CLR @SADRZ ; SET UP ADDRESS
MOV $UBIAS,@SADRX ; ...

; DETERMINE IF ROOM EXISTS IN THIS BLOCK

20$: SWTCH1 #NNTLUB ; POINT AT INPUT BUFFER
GETAD ; READ IN BLOCK FROM NETWORK POOL
MOV R2,R0 ; COPY VIRTUAL ADDRESS
ADD #100,R0 ; SET 32-WORD BLOCK VIRTUAL ADDRESS LIMIT
MOVB R2,R3 ; COPY VIRTUAL ADDRESS AGAIN
TST (R3)+ ; SKIP NEXT UNLOAD BLOCK POINTER
30$: CMP R3,R0 ; IS IT END OF THIS 32-WORD BLOCK?
BEQ 50$ ; IF EQ, YES
TST (R3)+ ; SKIP PDV INDEX ENTRY
TST (R3)+ ; SKIP LINE TABLE ADDRESS ENTRY
BEQ 40$ ; IF ZERO, THE REST OF THIS BLOCK IS AVAILABLE
BIT #1,(R3)+ ; ANY ALLOCATIONS THIS ENTRY?
BEQ 30$ ; IF EQ, NO
MOVB (R3)+,R4 ; GET ALLOCATION COUNT
MOVB (R3)+,R5 ; AND LIBRARY COUNT
ADD R4,R5 ; ADD THEM TOGETHER
ASL R5 ; MULTIPLY BY 4 (2 WORDS PER ENTRY)
ASL R5 ; ...

```

```

65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101 000004
102 000004
103
104
105
106
107
108 000004 012700 000000G
109 000010 016767 000000G 000000G
110 000016 000404
111
112 000020 012700 000000G
113
114 000024 005067 000000G
115 000030
116 000032 016703 000000G
117 000036 012767 000004 000000G
118 000044 010246
119 000046 010146
120 000050 010046
121 000052 005740

.SBTTL .ALLPR - ALLOCATE FROM PROCESS SPACE
.SBTTL .ALOCB - ALLOCATE FROM DSR
**-.ALLPR/.ALOCB/.ALOC1
THESE ROUTINES ALLOCATE 16-BIT STORAGE FROM THE MEMORY
DESCRIBED BY THE RESPECTIVE LISTHEAD:
$PRAVL - ALLOCATE FROM THE PROCESS/NTPOOL FREE SPACE
$CRAVL - ALLOCATE FROM COMM EXEC FREE SPACE
$CEAVL - ALLOCATE FROM THE SYSTEM'S DYNAMIC MEMORY

INPUTS:
R0 - ADDRESS OF LISTHEAD ( CALL AT .ALOC1 )
R1 - SIZE OF THE CORE BUFFER TO ALLOCATE IN BYTES.
$BIAS - BIAS OF PROCESS FOR PARTITION ALLOCATION
$PDVA - PDV OF PROCESS FOR PARTITION ALLOCATION

OUTPUTS:
C=1 IF INSUFFICIENT CORE IS AVAILABLE TO ALLOCATE THE BLOCK.
C=0 IF THE BLOCK IS ALLOCATED.
R0 - VIRTUAL ADDRESS OF THE ALLOCATED BLOCK
R2, R3, R4, R5 ARE PRESERVED

NOTE: THIS ROUTINE ASSUMES AN IN CORE LISTHEAD ( 3 WORDS ). THIS
SPEEDS ALLOCATION ( AND DEALLOCATION ) UP CONSIDERABLY. THE
MODULES VIMI & VIMO MAINTAIN THE LISTHEAD IMAGES. THESE
ROUTINES WILL NOT ALLOCATE UNLESS THE COMEXEC IS LOADED. THE
CONTEXT OF THE FE.CEX BIT HAS A SLIGHTLY CHANGED MEANING IN
VNP. IF THE BIT IS SET, CEX IMAGES ARE CORE RESIDENT, IF
CLEAR, THEN THE IMAGES ARE STILL ON THE DISK OR THE COMEXEC
IS NOT LOADED.

.MCALL GETRV,PUTRC,ERMSG$,ERRPT$,ERBLK$,GETAD,PCBDF$,PUTAD
.MCALL SWITCHI,SWTCHO,PKTDF$,RESRG,SAVRG

PCBDF$ ; PCB OFFSETS
PKTDF$ ; I/O PACKET OFFSETS
.ENABL LC

; ALLOCATION WILL BE FROM PARTITION IF ENTERED HERE
.ALLPR::MOV $PRAVL,R0 ; ADDR OF PROCESS FREESPACE LISTHEAD
MOV $BIAS,$RBLCK ; BIAS OF PROCESS
BR .ALOC2 ; ALLOCATE
.ALOCB::MOV $CRAVL,R0 ; GET ADDRESS OF SCOM LISTHEAD
.ALOC1::CLR $RBLCK ; WHEN CLEAR ALLOCATE FROM POOL
.ALOC2::SAVRG <R3> ; SAVE REG ;[EMP01]
MOV $RBLCK,R3 ; SAVE BIAS ;[EMP01]
MOV #4,$RSIZE ; SET TRANSFER SIZE
MOV R2,-(SP) ; SAVE
MOV R1,-(SP) ; REGISTERS
MOV R0,-(SP) ; SAVE LISTHEAD ADDRESS ON THE STACK
TST -(R0) ; POINT AT ROUNDING FACTOR

```


VAL16 CREATED BY MACRO ON 29-JUN-85 AT 02:12 PAGE 2 B 13

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
CALL	6-135 6-159 6-175 7-243 7-259 7-290 7-309 7-324
ERBLK\$	#6-98
ERMSG\$	#6-98
ERRPT\$	#6-98
GETAD	#6-98 6-135 7-243 7-259 7-309
GETRV	#6-98
PCBDF\$	#6-98 6-101
PKTDF\$	#6-99 6-102
PUTAD	#6-98 6-159 6-175 7-290 7-324
PUTRC	#6-98
RAND	#6-135 6-135 #6-159 6-159 #6-175 6-175 #7-243 7-243 #7-259 7-259
	#7-290 7-290 #7-309 7-309 #7-324 7-324
RESRG	#6-99 6-191
RETURN	6-192 7-342
SAVRG	#6-99 6-115
SWTCHI	#6-99
SWTCHO	#6-99

.TITLE VBIO - VNP BLOCK I/O ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - LOGICAL BLOCK I/O ROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 10-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/Rsx V1.0

```

150 ; ASL R2 ; CONVERT TO BYTES
151 ; ; NOTE - ERROR IN RSX-11D/IAS DOCUMENTATION
152 000112 005001 CLR R1 ; TASK ALWAYS STARTS AT VIRTUAL 0
153 000114 000426 BR 10$ ; COMPUTE BUFFER SIZE
154
155 : RSX11M,RSX11S
156
157 5$: GREG$S R0 ; GET REGION PARAMETERS
158 000132 103425 BCS 100$ ; NO DIRECTIVE IN SYSTEM ?
159 000134 016701 000000G MOV $DSW,R1 ; GET TASK BASE ADDRESS (MAPPED=0,UNMP=PHY)
160 000140 016002 000002 MOV G,GRGS(R0),R2 ; GET SIZE IN 32. WORD BLOCKS
161 000006 .REPT 6
162 ASL R2 ; CONVERT TO BYTES
163 .ENDR
164 000160 005701 TST R1 ; IS THIS A MEMORY MANAGEMENT SYSTEM ?
165 000162 001003 BNE 10$ ; IF NE, THEN NO, THIS IS PHYSICAL ADDRESS
166 000164 010601 MOV SP,R1 ; GET TASK'S REAL BASE ADDRESS
167 000166 042701 017777 BIC #17777,R1 ; ...
168
169 10$: ADD R1,R2 ; TASK'S HIGHEST ADDRESS IN BYTES
170 000174 160002 SUB R0,R2 ; MINUS XXSTR = SIZE OF POOL IN BYTES
171 000176 010267 000002' MOV R2,$XXLEN ; STORE LENGTH IN ROOT
172 000202 005020 CLR (R0)+ ; THIS IS THE ONLY POOL SEGMENT
173 000204 010210 MOV R2,(R0) ; THIS IS ITS LENGTH
174 000206 100$: RETURN
175 .END
176 000001

```

```

266 .SBTTL $ALLPR - ALLOCATE POOL FROM PROCESS
267
268 ;+ $ALLPR - ALLOCATE SINGLE WORD ADDRESSABLE STORAGE FROM A PROCESS
269 ; PARTITION. THE LISTHEAD USED LOOKS LIKE THE FOLLOWING:
270 ; .WORD ROUNDING FACTOR
271 ; $PRAVL::WORD OFFSET TO FREE SPACE LISTHEAD IN PROCESS
272 ; .WORD 0
273
274 ; INPUTS:
275 ; R1 - ALLOCATION SIZE
276 ; $PDVA - PROCESS PDV ADDRESS
277 ; OUTPUTS:
278 ; C-BIT - SUCCESS/FAILURE
279 ; R0 - VIRTUAL ADDRESS OF ALLOCATION
280 ; -
281
282 001024 010346 $ALLPR::MOV R3, -(SP) ; SAVE R3
283 001026 016703 000000G MOV $PDVA, R3 ; PDV ADDRESS
284 001032 012700 000000G MOV # $PRAVL, R0 ; ADDR OF PROCESS FREE SPACE LISTHEAD
285 001036 016310 000000G MOV Z.AVL(R3), (R0) ; POSSIBLY SOME FREE SPACE
286 001042 CALL $ALLPR ; ATTEMPT TO ALLOCATE FROM PROCESS
287 001046 103411 BCS 40$ ; FAILED IF C-SET
288
289 ; UPDATE THE FREE SPACE POINTER IN THE PDV
290 ;
291 001050 016703 000000G MOV $PDVA, R3 ; GET THE PDV ADDRESS
292 001054 016763 000000G 000000G MOV $PRAVL, Z.AVL(R3) ; UPDATE THE FREE SPACE POINTER
293 001062 042700 160000 BIC #160000, R0 ; CLEAR APR BITS...JUST IN CASE
294 001066 052700 120000 BIS #120000, R0 ; SET MAPPING FOR KERNEL APRS
295
296 001072 012603 40$: MOV (SP)+, R3 ; RESTORE R3
297 001074 RETURN

```

[illegible]

```

324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349

```

```

; *
; MAX14 - CHECK FOR NO MORE THAN 14. ALLOCATIONS
;
; INPUTS:
;   $NALL = NUMBER OF ALLOCATIONS
;   $NLIB = NUMBER OF LIBRARIES
;
; OUTPUTS:
;   C-BIT = SUCCESS/FAILURE
;   RO = DESTROYED
;
; -
;
MAX14:
    MOV    $NALL,RO      ; GET NUMBER OF ALLOCATIONS
    MOV    RO,-(SP)      ; SAVE IT
    MOV    $NLIB,RO      ; GET NUMBER OF LIBRARIES
    ADD    (SP),RO       ; ADD THE TWO
    CMP    #13,RO        ; ARE WE AT THE MAXIMUM ALREADY ?
    BCS    101$          ; IF CS, NO MORE ALLOWED
    RETURN
;
; ERROR CONDITION
;
101$:  MSG$R  KY          ; ONLY 14. ALLOCATIONS/LIBRARIES ALLOWED

```

```

000502 116700 000000G
000506 010046
000510 116700 000000G
000514 062600
000516 022700 000015
000522 103401
000524

```

```
VV      VV      AAAAAA      CCCCCCCC      333333
VV      VV      AAAAAA      CCCCCCCC      333333
VV      VV      AA          AA      CC      33      33
VV      VV      AA          AA      CC      33      33
VV      VV      AA          AA      CC      33      33
VV      VV      AA          AA      CC      33      33
VV      VV      AA          AA      CC      33      33
VV      VV      AA          AA      CC      33      33
VV      VV      AAAAAAAAAA      CC      33      33
VV      VV      AAAAAAAAAA      CC      33      33
VV      VV      AA          AA      CC      33      33
VV      VV      AA          AA      CC      33      33
VV      VV      AA          AA      CCCCCCCC      333333
VV      VV      AA          AA      CCCCCCCC      333333
```

```

LL          SSSSSSSS  TTTTTTTTTT
LL          SSSSSSSS  TTTTTTTTTT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SSSSSS    TT
LL          SSSSSS    TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LLLLLLLLLL SSSSSSSS  TT
LLLLLLLLLL SSSSSSSS  TT

```

```

103
104
105
106
107
108
109
110
111
112
113
114
115
116
117 000000
118 000000
119
120 000004
121 000004 032767 000000G 000000G
122 000012 001531
123 000014 005000
124 000016 032767 000000G 000000G
125 000024 001042
126
127 000026 005767 000000G
128 000032 100410
129
130 000034 026767 000000G 000000G
131 000042 001412
132 000044
133 000052 000406
134
135 000054 005767 000000G 5$:
136
137 000060 001403
138 000062
139 000070
140 000074 103476
141 000076 016701 000000G
142 000102 032767 000000G 000000G
143 000110 001003
144 000112
145 000116 000402
146 000120
147 000124 103465
148 000126 105267 000000G
149 000132 016702 000000G
150 000136 112722 000000G
151 000142 110022
152 000144 000300
153 000146 110022
154
155 000150 005767 000000G
156 000154 100404
157
158 000156 116722 000000G
159 000162 116722 000001G

* $BINF - ".BIN" ACTION ROUTINE
:
: INPUTS:
:   $BFALL=ALLOCATION SIZE
:   $BFLBN=LOGICAL BLOCK NUMBER
:   $BFPTR=BINARY BUFFER POINTER
:   $BFXFR=TRANSFER SIZE IN BYTES
:
: OUTPUTS:
:   NONE
:
$BING::
CALL $CORE2 ; PROCESS OPERAND 2

$BINF::
BIT #F2.FIL,$FLAG2 ; WAS VALID FILE NAME SEEN?
BEQ 30$ ; IF EQ, NO
CLR R0 ; NO ALLOCATION ADDRESS YET
BIT #F1.CHA,$FLAGS ; MUX CHARACTERISTICS UPDATE ONLY ?
BNE 13$ ; YES ..

TST .MGMGE ; MEMORY MANAGEMENT SYSTEM ?
BMI 5$ ; NO

CMP $BFBAS,$VALUE ; BUILT FOR CORRECT APR?
BEQ 10$ ; YES
EMSG$ ZA ; PRINT WARNING MESSAGE
BR 10$

TST $BFBAS ; BUILT FOR BASE ADDRESS OF 0?
BEQ 10$ ; IF EQ, YES
EMSG$ 76 ; ELSE PRINT WARNING MESSAGE
CALL MAX14 ; ALLOW ONLY 14 ALLOCATIONS
BCS 20$ ; IF CS, NO MORE ALLOWED
MOV $BFALL,R1 ; GET ALLOCATION SIZE NEEDED
BIT #F1.UMR,$FLAG1 ; IS UMR MAPPING NEEDED ?
BNE 11$ ; IF NE, YES
CALL $ALN18 ; ALLOCATE FROM NON-MAPPED AREA
BR 12$

CALL $ALL18 ; TRY TO ALLOCATE
BCS 10$ ; IF CS, FAILURE
INCB $NALL ; TALLY SUCCESSFUL ALLOCATION
MOV $BFPTR,R2 ; RESTORE BINARY BUFFER POINTER
MOVB #,$CORE,(R2)+ ; STORE FUNCTION CODE
MOVB R0,(R2)+ ; FIRST WORD OF ADDRESS
SWAB R0 ;
MOVB R0,(R2)+ ;

TST .MGMGE ; MEMORY MANAGEMENT SYSTEM ?
BMI 15$ ; NO

MOVB $VALUE,(R2)+ ; SECOND WORD OF ADDRESS
MOV$ $VALUE+1,(R2)+ ;

```


VAC4 CREATED BY MACRO ON 29-JUN-85 AT 02:09 PAGE 2 C 5
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
\$IOSB	= ***** GX	6-179
\$LBN	= ***** GX	*6-174
\$LLEN	= ***** GX	*6-176
\$NALL	= ***** GX	*6-148 *9-308 *10-366 11-387
\$NLIB	= ***** GX	11-389
\$PAD	= ***** GX	13-472
\$R	= ***** GX	*13-447 *13-456 *13-475
\$RBLCK	= ***** GX	*13-446 *13-457
\$READX	= ***** GX	6-177
\$REP.C	= ***** GX	5-88 5-89 5-91 5-93 12-413
\$RO5	= ***** GX	9-289 10-350
\$RO7	= ***** GX	9-286
\$RSIZE	= ***** GX	*13-450 *13-458 13-467 *13-468 13-469 *13-471 13-473 13-475 13-476
\$SCOM	= 000720 RG	#10-348
\$SIZE	= ***** GX	*6-167 *6-170 *9-326 *9-329 *10-373
\$TERR	= ***** GX	5-88 5-92 5-93
\$TWARN	= ***** GX	5-89 5-90 5-91
\$VALJE	= ***** GX	6-130 6-158 6-159 7-206 *7-213 7-217 8-236 *8-258 9-317
		9-318 *10-355 10-356 *10-362 10-371 10-372
\$VFLV1	= ***** GX	9-281 10-349
.BASEB	= ***** GX	5-75 5-81 7-213
.CORRE	= ***** GX	6-150 9-309
.MGMGE	= ***** GX	6-127 6-155 6-164 7-203 8-233 9-283 9-314 9-323 13-453
.SCOM	= ***** GX	10-367

VAC6 - VNP TEMPLATE ACTION ROUT MACRO V05.03b Saturday 29-Jun-85 02:10 ^{C 6}
Table of contents

5-	54	MACRO DEFINITIONS
6-	61	CONSTANT DEFINITIONS
7-	73	LOCAL DATA
13-	300	\$X2CHW - .X2CHW ACTION ROUTINE
13-	301	\$X3CHW - .X3CHW ACTION ROUTINE
14-	342	\$X2CHB - .X2CHB ACTION ROUTINE
14-	343	\$X3CHB - .X3CHB ACTION ROUTINE
16-	399	\$VXLLC - VERIFY NOT AN LLC

\$X3CHB - .X3CHB ACTION ROUTINE

```

384
385
386
387
388
389
390
391
392
393
394
395
396
397
001022
001022 032767 000000G 000000G
001030 001003
001032
001040

;+
; $VFLLC - VERIFY LLC ACTION ROUTINE
;
; INPUTS:
;     NONE
;
; OUTPUTS:
;     ERROR UNLESS LLC BEING LOADED
;-
$VFLLC:
    BIT    #FL.LLC,$FLAGS ; IS THIS AN LLC?
    BNE    10$             ; IF NE, YES
    MSG$   89              ; LLC OPERATOR IN DDM/DLC TEMPLATE
10$:      RETURN

```

\$VXLLC - VERIFY NOT AN LLC

```

224      ;+
225      ; BASE - CHECK ALREADY RESIDENT LIBRARY FOR CORRECT BASE ADDRESS
226      ;
227      ; INPUTS:
228      ; $LBIAS,$LBADR=DESCRIPTOR ADDRESS
229      ; $CAPR=USER'S APR NUMBER
230      ;
231      ; OUTPUTS:
232      ; NONE
233      ;
234      000402 016700 000000G      BASE:  MOV    $LBIAS,R0      ; GET DESCRIPTOR BIAS
235      000406      CALL    $GETLB      ; GET LIBRARY BLOCK
236      000412 066701 000000G      ADD    $LBADR,R1      ; OFFSET TO DESCRIPTOR
237      000416 116167 000000G 000000G  MOVB   LB:APR(R1),$BFBAS ; SAVE BASE APR
238      000424 126767 000000G 000000G 10$:  CMPB   $BFBAS,$CAPR      ; BUILT FOR CORRECT APR?
239      000432 001403      BEQ    20$      ; IF EQ, YES
240      000434      MSG$    V2      ; PRINT WARNING MESSAGE
241      000442      20$:    RETURN

```

VAC8 CREATED BY MACRO ON 29-JUN-85 AT 02:11 PAGE 1 C 9

SYMBOL CROSS REFERENCE

CREF 04.00

SYMBOL	VALUE	REFERENCES
FMT2	= ***** GX	5-71
FM.2	= 000000	#5-71
LBLK	= ***** GX	*6-94
\$\$\$BAS	= *****	5-71
\$ALB	= 000000 RG	#6-87
\$ALN18	= ***** GX	6-92
\$ELINE	= ***** GX	5-71
\$ERRY1	= 000000 R	#5-71
\$LBADR	= ***** GX	*6-96
\$LBERR	= ***** GX	6-112
\$LBIAS	= ***** GX	*6-95
\$PUTLB	= ***** GX	6-105
\$REP.C	= ***** GX	5-71
\$TERR	= ***** GX	5-71
.LBIAS	= ***** GX	*6-104
.LBUFF	= ***** GX	6-97
.MGMGE	= ***** GX	6-89

VALT - VNP LINE TABLE ALLOCATIO MACRO V05.03b Monday 08-Jul-85 22:06 ^{C 10}
Table of contents

7- 237 \$ALICB - ALLOCATE ICB

```

567 001036 060503      ADD    R5,R3      ; SKIP THE ALLOC/LIBR ENTRIES
568 001040 000761      BR     30$      ;
569 001042 024343      40$: CMP    -(R3),-(R3) ; POINT BACK AT THE ZERO WORD
570 001044 010304      MOV    R3,R4      ; COPY VIRTUAL ADDRESS
571 001046 066704 000000G ADD    $LDA,R4      ; INCLUDE UNLOAD SIZE NEEDED
572 001052 020400      CMP    R4,R0      ; IS THERE ENOUGH SPACE IN THIS BLOCK?
573 001054 101003      BHI    50$      ; IF HI, NO
574 001056      CALL    UBLTA      ; STORE LINE TABLE ADDRESS IN UNLOAD BLOCK
575 001062 000504      BR     100$     ; ALL DONE
576
577      ; LINK DOWN LIST TO NEXT UNLOAD BLOCK
578
579 001064 016767 000000G 000000G 50$: MOV    $NTLUB,$UBIAS ; SET VIRTUAL ADDRESS OF NEXT BLOCK
580 001072 001406      BEQ    70$      ; NO MORE, GO ALLOCATE ONE
581 001074 016777 000000G 000000G      MOV    $NTLUB,@$ADRZ ; SET UP ADDRESS
582 001102 005077 000000G      CLR    @$ADRZ ;
583 001106 000724      BR     20$      ; GO SEE IF ROOM EXISTS IN IT
584
585      ; ALLOCATE AN UNLOAD BLOCK SINCE NO ROOM EXIST IN ANY CURRENTLY ALLOCATED
586
587 70$:
588 001110      MOV    $UNBSZ,R1 ; SET SIZE OF ALLOCATION
589 001110 016701 000000G      SWTCHI ; RESTORE DEFAULT BUFFER ADDRESSES
590 001114      SWTCHO
591 001122      CALL    $ALN18 ; ALLOCATE AN UNLOAD BLOCK FROM NTPool
592 001130      BCS    101$ ; THIS IS VERY BAD !!
593 001134 103463      MCV    R0,$UBIAS ; SET ADDRESS
594 001136 010067 000000G      CLR    @$ADRZ ; 0 VIRTUAL ADDRESS
595 001142 005077 000000G      MOV    R0,@$ADRZ ; SET BLOCK BIAS
596 001146 010077 000000G      SWTCHI ; POINT AT INPUT BUFFER
597 001152      GETAD ; READ IN BLOCK IMAGE
598 001160      MOV    #$NTLUB,R3 ; POINT AT IT
599 001164 012703 000000G      CLR    (R3)+ ; CLEAR LINK TO NEXT UNLOAD BLOCK
600 001170 005023      MOV    R3,R4 ; ZERO THE REMAINDER OF THE 32. WORD BLOCK
601 001172 010304      MOV    #31,R5 ;
602 001174 012705 000037      CLR    (R4)+ ;
603 001200 005024      DEC    R5 ;
604 001202 005305      BNE    75$      ;
605 001204 001375      CALL    UBLTA ; STORE LINE TABLE ADDRESS DATA
606 001206      MOV    -CXUNL+$NTLHB,$BFR ; DO ANY PREVIOUS UNLOAD BLOCKS EXIST ?
607 001212 016767 000000G 000000G      BNE    90$      ; YES
608 001220 001003      MOV    R0,CXUNL+$NTLHB ; SET UP THIS BLOCK AS THE FIRST
609 001222 010067 000000G      BR     100$     ; AND WE'RE DONE
610 001226 000422
611
612      ; FIND LAST BLOCK AND ADD THIS UNLOAD BLOCK TO CHAIN
613
614 90$:
615 001230      SWTCHI ; $BFR ; USE OVERUSED BUFFER
616 001236 005077 000000G      CLR    @$ADRZ ; SET UP ADDRESS
617
618 001242 016777 000000G 000000G 91$: MOV    $BFR,@$ADRZ ;
619 001250      GETAD ; READ BLOCK INTO BUFFER
620 001254 005767 000000G      TST    $BFR ; IS THIS THE LAST BLOCK
621 001260 001370      BNE    91$      ; NOT YET ...
622 001262 016767 000000G 000000G      MOV    $UBIAS,$BFR ; LINK IN RECENTLY ALLOCATED UNLOAD BLOCK
623 001270      PUTAD ; REWRITE BLOCK

```

```

122 000054 061001      ADD      (R0),R1      ; ROUND ALLOCATION
123 000056 042001      BIC      (R0)+,R1    ; UP...
124
125 000060 000261      10$: SEC          ; ASSUME ZERO LENGTH BLOCK
126 000062 001446      BEQ      50$        ; WAS ZERO LENGTH BLOCK !
127 000064 011067 000000G MOV      (R0), $BFR    ; PREPARE TO ENTER LOOP
128
129 000070 010002      20$: MOV      R0,R2    ; SAVE ADDRESS OF CURRENT BLOCK
130 000072 016767 000002G 177700 MOV    $BFR+2,BLSIZ  ; SAVE SIZE OF PREVIOUS BLOCK
131 000100 016700 000000G MOV    $BFR,R0    ; GET ADDRESS OF NEXT BLOCK
132 000104 001765      BEQ      10$        ; IF EQ END OF FREE BLOCK CHAIN
133
134          .IF NDF R$$MPL                      ;[MP01]
135 00010F      GETAD  R0      ; READ FIRST TWO WORDS OF NEXT BLOCK
136          .IFF                                ;[MP01]
137
138          TST      R3      ; ALLOCATE FROM PROCESS SPACE ?      ;[MP01]
139          BEQ      22$      ; IF NO - BRANCH                    ;[MP01]
140          GETAD  R0      ; READ FROM PROCESS SPACE              ;[MP01]
141          BR       23$      ; CONTINUE                          ;[MP01]
142          22$:      GETRV  R0      ; DO A POSSIBLE D-SPACE READ ;[MP01]
143          .ENDC
144          23$:      CMP     R1,$BFR+2    ; BLOCK BIG ENOUGH?
145          BHI     20$      ; IF HI NO
146          SUB     R1,$BFR+2    ; REDUCE SIZE OF FREE BLOCK
147          BNE     30$      ; IF NE MORE LEFT IN BLOCK
148          CMP     R2,(SP)     ; IS THIS BEING REWRITTEN TO THE LISTHEAD ?
149          BNE     25$      ; NO
150          MOV     $BFR,(R2)    ; UPDATE APPROPRIATE LISTHEAD
151          BR      40$
152          000142 000414
153          ; SPECIAL CASE WHERE ALLOCATE BLOCK IS THE SAME SIZE AS THE FREE BLOCK
154
155          000144 016767 177630 000002G 25$: MOV    BLSIZ,$BFR+2 ; SET SIZE OF CURRENT BLOCK
156
157          .IF NDF R$$MPL                      ;[MP01]
158          PUTAD  R2      ; REMOVE BLOCK FROM LIST
159          .IFF                                ;[MP01]
160
161          TST      R3      ; ALLOCATE FROM PROCES SPACE ?      ;[MP01]
162          BFQ     26$      ; IF NO - BRANCH                    ;[MP01]
163          PUTAD  R2      ; ALLOCATE FROM PROCESS                ;[MP01]
164          BR      50$      ; CONTINUE                          ;[MP01]
165
166          26$:      PUTRC  R2      ; POSSIBLE D-SPACE READ      ;[MP01]
167          .ENDC
168
169          BR      50$
170          000162 000406      30$:
171          000164
172          .IF NDF R$$MPL                      ;[MP01]
173
174          PUTAD  R0      ; WRITE UPDATED BLOCK DESCRIPTOR
175          .IFF                                ;[MP01]
176          TST      R3      ; ALLOCATE FROM PROCESS SPACE      ;[MP01]
177          BEQ     35$      ; IF NO - BRANCH                    ;[MP01]
178

```


[illegible]

```

LL          SSSSSSSS  TTTTTTTTTT
LL          SSSSSSSS  TTTTTTTTTT
          SS          TT
LL          SS         TT
LL          SS         TT
LL          SS         TT
LL          SS         TT
          SSSSSS      TT
          SSSSSS      TT
          SS          TT
LL          SS         TT
LL          SS         TT
LL          SS         TT
LL          SS         TT
LLLLLLLLLLL SSSSSSSS  TT
LLLLLLLLLLL SSSSSSSS  TT

```

```

56      ;***
57      ; LIBRARY MACROS
58      ;***
59      .MCALL DIR$,Q10W$,GETRV,PUTRC,GETAD,PUTAD,ASR$
60
61      .ENABL LC
62
63      ;***
64      ; LOCAL DATA
65      ;***
66
67      000000      .PSECT DATA,D
68
69      ;
70      ; STATISTICS BLOCK
71
72      000000      STAT: .BLKW 5
73
74      ;
75      ; READ LOGICAL BLOCK DPB
76      ;
77      000012      003      014      LBNDPB: .BYTE 3,12.      ; DIC, LENGTH
78      000014      000000      .WORD 10,RLB      ; FUNCTION CODE
79      000016      000001      .WORD 1      ; LOGICAL UNIT
80      000020      000001      .WORD 1      ; EVENT FLAG
81      000022      000042      .WORD $IOSB      ; I/O STATUS BLOCK
82      000024      000000      .WORD 0      ; AST
83      000026      000000      $LBUF: .WORD 0      ; BUFFER ADDRESS
84      000030      000000      $LLEN: .WORD 0      ; BUFFER LENGTH
85      000032      000000      .WORD 0      ; CARRIAGE CONTROL
86      000034      .BLKW 2      $LBN: .BLKW 2      ; LOGICAL BLOCK NUMBER
87      000040      000000      .WORD 0
88
89      ;
90      ; I/O STATUS BLOCK
91
92      000042      $IOSB: .BLKW 2
93
94      000000      .PSECT

```

Symbol table

A\$\$CHK= 000000	G\$\$WRD= 000000	G.TSVL= 000030	M\$\$OVR= 000000	R\$\$K11= 000001
A\$\$CPS= 000000	G.RGFW= 000004	I\$\$RAR= 000000	N\$\$ACC= 000001	R\$\$SND= 000000
A\$\$PRI= 000000	G.RGRB= 000000	I\$\$RDN= 000000	N\$\$BUF= 000001	R\$\$11M= 000000
A\$\$TRP= 000000	G.RGRS= 000002	K\$\$CNT= 177546	N\$\$LDV= 000001	S\$\$WRG= 000000
C\$\$CKP= 000000	G.TSDU= 000036	K\$\$CSR= 177546	N\$\$MCP= 000001	S\$\$YSZ= 007600
C\$\$ORE= 000400	G.TSFV= 000024	K\$\$LDC= 000000	N\$\$MLL= 000001	T\$\$KMG= 000000
C\$\$RSH= 177564	G.TSGC= 000017	K\$\$TPS= 000074	N\$\$MOV= 000010	T\$\$MIN= 000000
D\$\$BUG= 177514	G.TSMT= 000022	LD\$LP= 000000	N\$\$NCT= 000001	V\$\$CTR= 001000
D\$\$ISK= 000000	G.TSNL= 000020	L\$\$ASG= 000000	N\$\$PEM= 000001	XXSTR= 000000R 003
D\$\$L11= 000001	G.TSPC= 000016	L\$\$DRV= 000000	P\$\$P45= 000000	X\$\$DBT= 000000
D\$\$YNC= 000000	G.TSPN= 000004	L\$\$P11= 000001	P\$\$WRD= 000000	\$DSW = ***** GX
D\$\$YNM= 000000	G.TSPR= 000014	L\$\$11R= 000000	Q\$\$OPT= 000010	\$SAVAL= ***** GX
E\$\$XPR= 000000	G.TSRN= 000010	M\$\$CRB= 000124	RX\$IAS= 000003	\$XXAVL 000006RG 002
F\$\$LVL= 000001	G.TSSY= 000034	M\$\$CRX= 000000	RX\$11D= 000000	\$XXBUF 000000RG 002
G\$\$TTP= 000000	G.TSTN= 000000	M\$\$FCS= 000000	RX\$11M= 000001	\$XXIN1 000000RG
G\$\$TSS= 000000	G.TSTS= 000032	M\$\$MGE= 000000	RX\$11S= 000002	\$XXLEN 000002RG 002
G\$\$TTK= 000000	G.TSVA= 000026	M\$\$NET= 000000	R\$\$DER= 000000	\$\$\$OST= 000006

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
 000014 001 (RW,I,LCL,REL,CON)
 DATA 000012 002 (RW,D,LCL,REL,CON)
 .POOL 000210 003 (RW,I,LCL,REL,OVR)
 Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 9613 Words (38 Pages)
 Size of core pool: 14440 Words (55 Pages)
 Operating system: RSX-11M/PLUS

Elapsed time: 00:00:08.31
 SY:VBUF.V2,[132,134]VBUF/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VBUF

```

299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314 001076
315 001076 032767 000000G 000000G
316 001104 001405
317
318
319 001106 012700 000000G
320 001112
321 001116 103002
322
323
324 001120
325 001120
326 001124

.SBTTL $ALL16 - ALLOCATE POOL FROM DSR
;+
;$ALL16 - ALLOCATE SINGLEWORD ADDRESSABLE STORAGE ( SCOM )
;
;INPUTS:
;R1 - BYTE COUNT NEEDED
;
;OUTPUTS:
;C-BIT - SUCCESS/FAILURE
;R0 - BUFFER ADDRESS
;ALL OTHER REGISTERS PRESERVED
;-

.ENABL LSB

$ALL16::
BIT #FE,CEX,$FMASK ; IS COMEXEC LOADED ?
BEQ $ALL15 ; NO, THEN VNP ALLOCATIONS FAIL

;IF NDF,R$$MPL
MOV #SCEAVL,R0 ; GET CEX ALLOCATION POINTER
CALL .ALOC1 ; TRY TO ALLOCATE FROM CEX POOL
BCC 10$ ; IF CC, SUCCESS
.ENDC

$ALL15::
CALL .ALOCB ; ATTEMPT ALLOCATION FROM RSX POOL
10$: RETURN

```

```

VV      VV      AAAAAA      CCCCCCCC      222222
VV      VV      AAAAAA      CCCCCCCC      222222
VV      VV      AA        AA      CC      22      22
VV      VV      AA        AA      CC      22      22
VV      VV      AA        AA      CC      22      22
VV      VV      AA        AA      CC      22      22
VV      VV      AA        AA      CC      22      22
VV      VV      AA        AA      CC      22      22
VV      VV      AAAAAAAA      CC      22      22
VV      VV      AAAAAAAA      CC      22      22
VV      VV      AA        AA      CC      22      22
VV      VV      AA        AA      CC      22      22
VV      VV      AA        AA      CCCCCCCC      2222222222
VV      VV      AA        AA      CCCCCCCC      2222222222

```

```

....
....
....
....

```

```

LL      SSSSSSSS      TTTTTTTTTT
LL      SSSSSSSS      TTTTTTTTTT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SSSSSS      TT
LL      SSSSSS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LLLLLLLLLL      SSSSSSSS      TT
LLLLLLLLLL      SSSSSSSS      TT

```

```

351      ;+
352      $MPTB - ''MPTAB'' ACTION ROUTINE
353      $LFILE - ''LFILE'' ACTION ROUTINE
354
355      INPUTS:
356      $SMTP/$VALUE=REPEAT COUNT
357      $FILE=SECONDARY TEMPLATE NAME
358
359      OUTPUTS:
360      R0,R1,R2=DESTROYED
361      -
362
363      .ENABL  LSB
364
365      $MPTB::
366      000534      CALL    $VFDLC      ; MAKE SURE IT'S A DLC
367      000534      MOV     #F1.MPT,R0 ; MAKE SURE THIS IS THE ONLY ONE
368      000540      CALL    $VFSPC
369      000540      BCS     40$          ; IF CS, "ERROR
370      000550      103500      BIT     #F1.MPL,$FLAG1 ; HAS ''MPLHD'' BEEN SEEN?
371      000552      032767      BEQ     101$          ; IF EQ, NO
372      000560      001475      MOV     #.MPTAB,R0    ; SET FUNCTION CODE
373      000562      012700      MOV     $MPLD,R1      ; GET ''MPLHD'' DISPLACEMENT
374      000566      016701      MOV     $SMTP,$VALUE ; DO IT <STATION COUNT> TIMES
375      000572      016767      BR      10$
376      000602      $LFILE::
377      000602      012700      MOV     #F1.LFI,R0    ; MAKE SURE THIS IS THE ONLY ONE
378      000606      CALL    $VFSPC
379      000612      103457      BCS     40$          ; IF CS, "ERROR
380      000614      032767      BIT     #F1.LFL,$FLAG1 ; HAS ''LFLHD'' BEEN SEEN?
381      000622      001457      BEQ     111$          ; IF EQ, YES
382      000624      012700      MOV     #.LFILE,R0    ; SET FUNCTION CODE
383      000630      016701      MOV     $LFLD,R1      ; GET ''LFLHD'' DISPLACEMENT
384      000634      022767      CMP     #256,$VALUE  ; IS COUNT > 256.?
385      000642      103452      BLO     CTLERR       ; IF LO, YES
386      000644      005767      TST     $VALUE       ; IS COUNT 0?
387      000650      001440      BEQ     40$          ; IF EQ, YES
388      000652      CALL    $RQ4              ; REQUEST BINARY FILE
389      000656      103435      BCS     40$          ; IF CS, NO ACCESS
390      000660      110022      MOVB    R0,(R2)+      ; STORE FUNCTION CODE
391      000662      116722      MOVB    $VALUE,(R2)+ ; FILE REPEAT COUNT
392      000666      110122      MOVB    R1,(R2)+      ; AND LISTHEAD DISPLACEMENT
393      000670      000301      SWAB    R1
394      000672      110112      MOVB    R1,(R2)
395      000674      016746      MOV     $SIZE,-(SP)   ; SAVE CURRENT LINE TABLE SIZE
396      000700      CALL    $FILE               ; PROCESS SECONDARY TEMPLATE
397      000704      CALL    $EVEN              ; AUTOMATIC WORD RE-ALIGNMENT
398      000710      012600      MOV     (SP)+,R0     ; RESTORE SIZE FROM START OF FILE
399      000712      CALL    $RQ1              ; REQUEST BINARY FILE AGAIN
400      000716      103415      BCS     40$          ; IF CS, NO ACCESS
401      000720      112712      MOVB    #.EOF,(R2)   ; STORE END OF FILE FUNCTION CODE
402      000724      166700      SUB     $SIZE,R0     ; GIVING AMOUNT ADDED BY SECONDARY TEMPLATE
403      000730      001005      BNE     30$
404      000732      EMSG$R      43
405      000740      160067      000000G      20$: SUB    R0,$SIZE
406      000744      005367      000000G      30$: DEC    $VALUE
407      000750      001373      BNE     20$

```

.TITLE VAC3 - VNP TEMPLATE ACTION ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - TEMPLATE SYNTAX TPARS ACTION ROUTINES, PART 3

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

```

160
161 00166 116722 000000G 15$: MOVB $BFALL,(R2) ; AND COUNT (FOR CLEANUP)
162 000172 116712 000001G MOVB $BFALL+1.(R2) ; ...
163
164 000176 005767 000000G TST .MGMGE ; MEMORY MANAGEMENT SYSTEM ?
165 000202 100404 BMJ 17$ ; NO
166
167 000204 062767 000004 000000G ADD #4,$SIZE ; ADJUST DATA BASE SIZE
168 000212 000403 BR 18$
169
170 000214 062767 000002 000000G 17$: ADD #2,$SIZE ; ADJUST DATA BASE SIZE
171
172 000222 000300 18$: SWAB R0 ; RESTORE ALLOCATION BIAS
173 000224 001424 BEQ 30$ ; IF ZERO, MUX CHARACTERISTICS UPDATE
174 000226 016767 000000G 000000G MOV $BFLBN,$LBN ; SET LOGICAL BLOCK NUMBER
175 000234 016767 000002G 000002G MOV $BFLBN+2,$LBN+2 ; ...
176 000242 016767 000000G 000000G MOV $BFXFR,$LLEN ; AND TRANSFER SIZE IN BYTES
177 000250 CALL $READX ; TRANSFER BINARY FILE INTO CORE
178 000254 103006 BCC 20$ ; IF CC, NO PROBLEM
179 000256 116777 000000G 000000G MOVB $IOSB,$CLIOS ; COPY ERROR CODE
180 000264 MSG$ 77 ; READ FAILURE
181 000272 20$: CALL $CLDEQ ; CLOSE BINARY FILE
182 000276 30$: RETURN
183
184 ; ERROR CONDITION
185
186
187 000300 01$: CALL $CLDEQ ; CLOSE BINARY FILE
188 000304 MSG$R 26 ; RESOURCE ALLOCATION FAILURE

```


VAC4 CREATED BY MACRO ON 29-JUN-85 AT 02:09 PAGE 3 D 5

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

ASL\$	#5-59	13-449							
ASR\$	#5-59	7-211							
CALL	6-118	6-132	6-138	6-139	6-144	6-177	6-180	6-181	6-187
	8-256	9-278	9-281	9-286	9-289	9-298	9-304	9-306	10-349
	10-350	10-358	10-363	10-365	12-413	13-472			
EMSG\$	#5-59	6-132	6-138	6-180	8-256				
EMSG\$R	#5-59	6-188	9-335	11-398					
NTLR\$	#5-59	5-88	5-89	5-90	5-91	5-92	5-93		
PUTAD	#5-59	13-472							
PUTRC	#5-59								
RAND	#13-472	13-472							
RETURN	6-182	7-215	7-218	8-261	9-330	10-374	11-393	12-422	13-481

.TITLE VAC6 - VNP TEMPLATE ACTION ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - TEMPLATE SYNTAX IPARS ACTION ROUTINES, PART 6

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLU V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-
DECnet- V4.2
DECnet- us V3.0
DECnet-M1 .RSX V1.0

```

399                                     .SBTTL $VXILC - VERIFY NOT AN LLC
400
401                                     ;+
402                                     ; $VXLLC - VERIFY NOT AN LLC
403                                     ;
404                                     ; INPUTS:
405                                     ; $FLAGS - FLAGS WORD
406                                     ;
407                                     ; OUTPUTS:
408                                     ; ERROR IF LLC BEING LOADED
409                                     ;
410                                     ;
411                                     ;
412 001042 032767 000000C 00C000G $VXLLC: BIT    #FL.DDM!FL.DLC,$FLAGS    ; IS IT DDM OR DLC?
413 001050 001003                                     BNE    10$                ; BR IF YES
414 001052                                     MSG$    90                ; ELSE ERROR
415 001060                                     10$:    RETURN
416
417                                     .END
000001

```

```

243                                     ;+
244                                     ; USE - INCREMENT LIBRARY USE COUNT
245                                     ;
246                                     ; INPUTS:
247                                     ; $LBIAS=DESCRIPTOR BIAS (MAPPE) ONLY)
248                                     ; $LBADR=DESCRIPTOR VIRTUAL ADDRESS
249                                     ;
250                                     ; OUTPUTS:
251                                     ; LB.USE, $NLIB INCREMENTED
252                                     ;
253 000444 016702 000000G      USE:  MOV  $BFPTR,R2      ; GET BINARY BUFFER POINTER
254 000450 112722 000000G      MOV  #.LIBR,(R2)+      ; STORE FUNCTION CODE
255 000454 016700 000000G      MOV  $LBIAS,R0          ; GET LIBRARY DESCRIPTOR BIAS
256 000460 110022              MOV  R0,(R2)+            ; STORE DESCRIPTOR BIAS
257 000462 116722              MOV  $LBIAS+1,(R2)+      ;
258 000466 116722 000000G      MOV  $LBADR,(R2)+      ; STORE DESCRIPTOR ADDRESS
259 000472 116722 000001G      MOV  $LBADR+1,(R2)+     ;
260 000476 005767 000000G      TST  .MGMGE            ; MAPPED TARGET SYSTEM?
261 000502 100410              BMI  10$                ; IF MI, NO
262 000504 116722 000000G      MOV  $VALUE,(R2)+       ; STORE USER'S MAPPING TO LIBRARY
263 000510 116712 000001G      MOV  $VALUE+1,(R2)      ;
264 000514 062767 000004 000000G  ADD  #4,$SIZE        ; ADJUST LINE TABLE SIZE
265 000522 000403              BR   20$                ;
266 000524 062767 000002 000000G 10$: ADD  #2,$SIZE        ; ADJUST LINE TABLE SIZE
267 000532 032767 000000G 000000G 20$: BIT  #FL.CHA,$FLAGS ; MUX CHARACTERISTICS UPDATE ONLY ?
268 000540 001012              BNE  30$                ; NO .. DON'T CHANGE USE COUNT
269 000542              CALL  $GETLB                    ; GET LIBRARY BLOCK
270 000546 066701 000000G      ADD  $LBADR,R1          ; OFFSET TO DESCRIPTOR
271 000552 005261 000000G      INC  LB.USE(R1)         ; INCREMENT USE COUNT
272 000556              CALL  $PUTLB                    ; RE-WRITE LIBRARY BLOCK
273 000562 105267 000000G      INCB $NLIB              ; INCREMENT LIBRARY COUNT
274 000566              30$:  RETURN

```

VAC8 CREATED BY MACRO ON 29-JUN-85 AT 02:11 PAGE 2 D 9
MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

CALL	6-92	6-105	6-112
EMSG\$R	#5-59	5-113	
NTLR\$	#5-59	5-71	
RETURN	6-107		

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50

.TITLE VALT - VNP LINE TABLE ALLOCATION ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - LINE TABLE ALLOCATION ROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

D 11

```

624
625 001274          100$: SWITCHI          ; RESET DEFAULT BUFFER
626 001302          RETURN
627
628                ;
629                ; ERROR CONDITION
630
631 001304 016700 000000G 101$: MOV    $$ADDR,R0      ; DE-ALLOCATE LINE TABLE
632 001310 016701 000000G    MOV    $SIZE,R1      ;
633 001314          CALL  $DEA16      ; ...
634 001320 005067 000000G    CLR    $$ADDR      ; INDICATE NO LINE TABLE
635 001324          MSG$R  86      ; UNLOAD BLOCK ALLOCATION FAILURE

```

E 11

.ALOCB - ALLOCATE FROM DSR

```

179                                     PUTAD   R0           ; READ FROM PROCESS           ;[MP01]
180                                     BR       36$          ; CONTINUE                     ;[MP01]
181                                     35$: PUTRC   R0           ; DO A POSSIBLE D-SPACE WRITE  ;[MP01]
182
183                                     .ENDC                                     ;[MP01]
184 000174                               36$:
185
186 000174 066700 000002G               40$: ADD     $BFR+2,R0      ; CALCULATE ADDRESS OF ALLOCATED BLOCK
187 000200 005226                               50$: INC     (SP)+      ; CLEAR LISTHEAD ADDRESS FROM STACK
188                                     ; WITHOUT AFFECTING THE C-BIT
189 000202 012601                               MOV     (SP)+,R1      ; RESTORE ORIGINAL REQUESTED SIZE
190 000204 012602                               MOV     (SP)+,R2      ; RESTORE R2
191 000206                               RESRG   <R3>          ; RESTORE REG                 ;[MP01]
192 000210
193

```

.DEARR - DEALLOCATE FROM PROCESS

VAL22 - VNP BLOCK ALLOCATION RO MACRO V05.03b Saturday 29-Jun-85 02:12 ^{E 13}
Table of contents

6- 111 .AMEM - ALLOCATE VIRTUAL MEMORY IN NTPool
7- 198 .DMEM - DEALLOCATE MEMORY FROM VIRTUAL NTPool

```

96
97
98
99
100
101
102
103
104
105
106
107 000000
108 000000 012700 000000'
109 000004
110 000010 012700 000002'
111 000014 011067 000036'
112 000020 014067 000034'
113 000024

;+
;$CLOPE - USE GCL TO OPEN A CONTIGUOUS FILE
;
;INPUTS:
;R1=FILE SPEC ADDRESS
;R2=FILE SPEC LENGTH
;
;OUTPUTS:
;R0=STATISTICS BLOCK ADDRESS
;$LBN=LABEL BLOCK LOGICAL BLOCK NUMBER
;-
;$CLOPE::
;MOV #STAT,R0 ; GET STATISTICS WHILE OPENING
;CALL $CLOST ; USE GCL TO OPEN THE FILE
;MOV #STAT+2,R0 ; GET STATISTICS BLOCK ADDRESS
;MOV (R0),$LBN+2 ; COPY LABEL BLOCK LBN
;MOV -(R0),$LBN ; ...
;RETURN

```

VBUF CREATED BY MACRO ON 29-JUN-85 AT 02:13 PAGE 1 D 15

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
G.RGRB	= 000000	6-157
G.RGRS	= 000002	6-160
G.TSSY	= 000034	6-159
G.TSTN	= 000000	6-137
G.TSTS	= 000032	6-149
XXSTR	= J00000 R	6-114 #6-119 6-120
\$DSW	= ***** GX	6-159
\$SAVAL	= ***** GX	6-119
\$XXAVL	000006 RG	#5-90 6-121
\$XXBUF	000000 RG	#5-83 6-112 *6-125
\$XXINI	000000 RG	#6-111
\$XXLEN	000002 RG	#5-84 *6-171

```

328      .SBTTL  $DEA16 - DEALLOCATE FROM DSR
329      ;
330      ; $DEA16 - DE-ALLOCATE SINGLEWORD ADDRESSABLE STORAGE ( SCOM )
331      ; $DEA16 - DE- ALLOCATE ICB FROM ICB POOL (M+ ONLY)
332      ;
333      ; INPUTS:
334      ; R0 -    BUFFER ADDRESS
335      ; R1 -    BYTE COUNT
336      ;
337      ; OUTPUTS:
338      ; C-BIT WILL BE SET IF BUFFER ADDRESS IS ILLEGAL.
339      ;
340      ;
341      .IF DF  R$$$MPL                                ;[MP01]
342
343      $DEA16::CALL  $$SAVAL                                ;[MP01]
344      MOV    #SICAVL,R3                                ;[MP01]
345      BR     DEAC                                        ;[MP01]
346      .ENDC                                           ;[MP01]
347
348      $DEA16::CALL  $$SAVAL                                ; SAVE ALL REGISTERS
349      001126
350      001126
351      001132  012703  000000G                          ; ASSUME BUFFER IS FROM RSX POOL
352
353      .IF  NDF,R$$$MPL
354      001136  026700  000000G                          ; IS BUFFER FROM RSX POOL?
355      001142  101007
356      001144  012703  000000G                          ; IF H1, YES
357      001150  000261
358      001152  032767  000000G 000000G                  ; ASSUME CEX POOL THEN
359      001160  001402
360
361      .IF  NDF,R$$$MPL
362      001162
363      001166
364
365      .IF  NDF,R$$$MPL
366      001168
367      001170
368      001172
369      001174
370      001176
371      001178
372      001180
373      001182
374      001184
375      001186
376      001188
377      001190
378      001192
379      001194
380      001196
381      001198
382      001200
383      001202
384      001204
385      001206
386      001208
387      001210
388      001212
389      001214
390      001216
391      001218
392      001220
393      001222
394      001224
395      001226
396      001228
397      001230
398      001232
399      001234
400      001236
401      001238
402      001240
403      001242
404      001244
405      001246
406      001248
407      001250
408      001252
409      001254
410      001256
411      001258
412      001260
413      001262
414      001264
415      001266
416      001268
417      001270
418      001272
419      001274
420      001276
421      001278
422      001280
423      001282
424      001284
425      001286
426      001288
427      001290
428      001292
429      001294
430      001296
431      001298
432      001300
433      001302
434      001304
435      001306
436      001308
437      001310
438      001312
439      001314
440      001316
441      001318
442      001320
443      001322
444      001324
445      001326
446      001328
447      001330
448      001332
449      001334
450      001336
451      001338
452      001340
453      001342
454      001344
455      001346
456      001348
457      001350
458      001352
459      001354
460      001356
461      001358
462      001360
463      001362
464      001364
465      001366
466      001368
467      001370
468      001372
469      001374
470      001376
471      001378
472      001380
473      001382
474      001384
475      001386
476      001388
477      001390
478      001392
479      001394
480      001396
481      001398
482      001400
483      001402
484      001404
485      001406
486      001408
487      001410
488      001412
489      001414
490      001416
491      001418
492      001420
493      001422
494      001424
495      001426
496      001428
497      001430
498      001432
499      001434
500      001436
501      001438
502      001440
503      001442
504      001444
505      001446
506      001448
507      001450
508      001452
509      001454
510      001456
511      001458
512      001460
513      001462
514      001464
515      001466
516      001468
517      001470
518      001472
519      001474
520      001476
521      001478
522      001480
523      001482
524      001484
525      001486
526      001488
527      001490
528      001492
529      001494
530      001496
531      001498
532      001500
533      001502
534      001504
535      001506
536      001508
537      001510
538      001512
539      001514
540      001516
541      001518
542      001520
543      001522
544      001524
545      001526
546      001528
547      001530
548      001532
549      001534
550      001536
551      001538
552      001540
553      001542
554      001544
555      001546
556      001548
557      001550
558      001552
559      001554
560      001556
561      001558
562      001560
563      001562
564      001564
565      001566
566      001568
567      001570
568      001572
569      001574
570      001576
571      001578
572      001580
573      001582
574      001584
575      001586
576      001588
577      001590
578      001592
579      001594
580      001596
581      001598
582      001600
583      001602
584      001604
585      001606
586      001608
587      001610
588      001612
589      001614
590      001616
591      001618
592      001620
593      001622
594      001624
595      001626
596      001628
597      001630
598      001632
599      001634
600      001636
601      001638
602      001640
603      001642
604      001644
605      001646
606      001648
607      001650
608      001652
609      001654
610      001656
611      001658
612      001660
613      001662
614      001664
615      001666
616      001668
617      001670
618      001672
619      001674
620      001676
621      001678
622      001680
623      001682
624      001684
625      001686
626      001688
627      001690
628      001692
629      001694
630      001696
631      001698
632      001700
633      001702
634      001704
635      001706
636      001708
637      001710
638      001712
639      001714
640      001716
641      001718
642      001720
643      001722
644      001724
645      001726
646      001728
647      001730
648      001732
649      001734
650      001736
651      001738
652      001740
653      001742
654      001744
655      001746
656      001748
657      001750
658      001752
659      001754
660      001756
661      001758
662      001760
663      001762
664      001764
665      001766
666      001768
667      001770
668      001772
669      001774
670      001776
671      001778
672      001780
673      001782
674      001784
675      001786
676      001788
677      001790
678      001792
679      001794
680      001796
681      001798
682      001800
683      001802
684      001804
685      001806
686      001808
687      001810
688      001812
689      001814
690      001816
691      001818
692      001820
693      001822
694      001824
695      001826
696      001828
697      001830
698      001832
699      001834
700      001836
701      001838
702      001840
703      001842
704      001844
705      001846
706      001848
707      001850
708      001852
709      001854
710      001856
711      001858
712      001860
713      001862
714      001864
715      001866
716      001868
717      001870
718      001872
719      001874
720      001876
721      001878
722      001880
723      001882
724      001884
725      001886
726      001888
727      001890
728      001892
729      001894
730      001896
731      001898
732      001900
733      001902
734      001904
735      001906
736      001908
737      001910
738      001912
739      001914
740      001916
741      001918
742      001920
743      001922
744      001924
745      001926
746      001928
747      001930
748      001932
749      001934
750      001936
751      001938
752      001940
753      001942
754      001944
755      001946
756      001948
757      001950
758      001952
759      001954
760      001956
761      001958
762      001960
763      001962
764      001964
765      001966
766      001968
767      001970
768      001972
769      001974
770      001976
771      001978
772      001980
773      001982
774      001984
775      001986
776      001988
777      001990
778      001992
779      001994
780      001996
781      001998
782      002000
783      002002
784      002004
785      002006
786      002008
787      002010
788      002012
789      002014
790      002016
791      002018
792      002020
793      002022
794      002024
795      002026
796      002028
797      002030
798      002032
799      002034
800      002036
801      002038
802      002040
803      002042
804      002044
805      002046
806      002048
807      002050
808      002052
809      002054
810      002056
811      002058
812      002060
813      002062
814      002064
815      002066
816      002068
817      002070
818      002072
819      002074
820      002076
821      002078
822      002080
823      002082
824      002084
825      002086
826      002088
827      002090
828      002092
829      002094
830      002096
831      002098
832      002100
833      002102
834      002104
835      002106
836      002108
837      002110
838      002112
839      002114
840      002116
841      002118
842      002120
843      002122
844      002124
845      002126
846      002128
847      002130
848      002132
849      002134
850      002136
851      002138
852      002140
853      002142
854      002144
855      002146
856      002148
857      002150
858      002152
859      002154
860      002156
861      002158
862      002160
863      002162
864      002164
865      002166
866      002168
867      002170
868      002172
869      002174
870      002176
871      002178
872      002180
873      002182
874      002184
875      002186
876      002188
877      002190
878      002192
879      002194
880      002196
881      002198
882      002200
883      002202
884      002204
885      002206
886      002208
887      002210
888      002212
889      002214
890      002216
891      002218
892      002220
893      002222
894      002224
895      002226
896      002228
897      002230
898      002232
899      002234
900      002236
901      002238
902      002240
903      002242
904      002244
905      002246
906      002248
907      002250
908      002252
909      002254
910      002256
911      002258
912      002260
913      002262
914      002264
915      002266
916      002268
917      002270
918      002272
919      002274
920      002276
921      002278
922      002280
923      002282
924      002284
925      002286
926      002288
927      002290
928      002292
929      002294
930      002296
931      002298
932      002300
933      002302
934      002304
935      002306
936      002308
937      002310
938      002312
939      002314
940      002316
941      002318
942      002320
943      002322
944      002324
945      002326
946      002328
947      002330
948      002332
949      002334
950      002336
951      002338
952      002340
953      002342
954      002344
955      002346
956      002348
957      002350
958      002352
959      002354
960      002356
961      002358
962      002360
963      002362
964      002364
965      002366
966      002368
967      002370
968      002372
969      002374
970      002376
971      002378
972      002380
973      002382
974      002384
975      002386
976      002388
977      002390
978      002392
979      002394
980      002396
981      002398
982      002400
983      002402
984      002404
985      002406
986      002408
987      002410
988      002412
989      002414
990      002416
991      002418
992      002420
993      002422
994      002424
995      002426
996      002428
997      002430
998      002432
999      002434
1000     002436
1001     002438
1002     002440
1003     002442
1004     002444
1005     002446
1006     002448
1007     002450
1008     002452
1009     002454
1010     002456
1011     002458
1012     002460
1013     002462
1014     002464
1015     002466
1016     002468
1017     002470
1018     002472
1019     002474
1020     002476
1021     002478
1022     002480
1023     002482
1024     002484
1025     002486
1026     002488
1027     002490
1028     002492
1029     002494
1030     002496
1031     002498
1032     002500
1033     002502
1034     002504
1035     002506
1036     002508
1037     002510
1038     002512
1039     002514
1040     002516
1041     002518
1042     002520
1043     002522
1044     002524
1045     002526
1046     002528
1047     002530
1048     002532
1049     002534
1050     002536
1051     002538
1052     002540
1053     002542
1054     002544
1055     002546
1056     002548
1057     002550
1058     002552
1059     002554
1060     002556
1061     002558
1062     002560
1063     002562
1064     002564
1065     002566
1066     002568
1067     002570
1068     002572
1069     002574
1070     002576
1071     002578
1072     002580
1073     002582
1074     002584
1075     002586
1076     002588
1077     002590
1078     002592
1079     002594
1080     002596
1081     002598
1082     002600
1083     002602
1084     002604
1085     002606
1086     002608
1087     002610
1088     002612
1089     002614
1090     002616
1091     002618
1092     002620
1093     002622
1094     002624
1095     002626
1096     002628
1097     002630
1098     002632
1099     002634
1100     002636
1101     002638
1102     002640
1103     002642
1104     002644
1105     002646
1106     002648
1107     002650
1108     002652
1109     002654
1110     002656
1111     002658
1112     002660
1113     002662
1114     002664
1115     002666
1116     002668
1117     002670
1118     002672
1119     002674
1120     002676
1121     002678
1122     002680
1123     002682
1124     002684
1125     002686
1126     002688
1127     002690
1128     002692
1129     002694
1130     002696
1131     002698
1132     002700
1133     002702
1134     002704
1135     002706
1136     002708
1137     002710
1138     002712
1139     002714
1140     002716
1141     002718
1142     002720
1143     002722
1144     002724
1145     002726
1146     002728
1147     002730
1148     002732
1149     002734
1150     002736
1151     002738
1152     002740
1153     002742
1154     002744
1155     002746
1156     002748
1157     002750
1158     002752
1159     002754
1160     002756
1161     002758
1162     002760
1163     002762
1164     002764
1165     002766
1166     002768
1167     002770
1168     002772
1169     002774
1170     002776
1171     002778
1172     002780
1173     002782
1174     002784
1175     002786
1176     002788
1177     002790
1178     002792
1179     002794
1180     002796
1181     002798
1182     002800
1183     002802
1184     002804
1185     002806
1186     002808
1187     002810
1188     002812
1189     002814
1190     002816
1191     002818
1192     002820
1193     002822
1194     002824
1195     002826
1196     002828
1197     002830
1198     002832
1199     002834
1200     002836
1201     002838
1202     002840
1203     002842
1204     002844
1205     002846
1206     002848
1207     002850
1208     002852
1209     002854
1210     002856
1211     002858
1212     002860
1213     002862
1214     002864
1215     002866
1216     002868
1217     002870
1218     002872
1219     002874
1220     002876
1221     002878
1222     002880
1223     002882
1224     002884
1225     002886
1226     002888
1227     002890
1228     002892
1229     002894
1230     002896
1231     002898
1232     002900
1233     002902
1234     002904
1235     002906
1236     002908
1237     002910
1238     002912
1239     002914
1240     002916
1241     002918
1242     002920
1243     002922
1244     002924
1245     002926
1246     002928
1247     002930
1248     002932
1249     002934
1250     002936
1251     002938
1252     002940
1253     002942
1254     002944
1255     002946
1256     002948
1257     002950
1258     002952
1259     002954
1260     002956
1261     002958
1262     002960
1263     002962
1264     002964
1265     002966
1266     002968
1267     002970
1268     002972
1269     002974
1270     002976
1271     002978
1272     002980
1273     002982
1274     002984
1275     002986
1276     002988
1277     002990
1278     002992
1279     002994
1280     002996
1281     002998
1282     003000
1283     003002
1284     003004
1285     003006
1286     003008
1287     003010
1288     003012
1289     003014
1290     003016
1291     003018
1292     003020
1293     003022
1294     003024
1295     003026
1296     003028
1297     003030
1298     003032
1299     003034
1300     003036
1301     003038
1302     003040
1303     003042
1304     003044
1305     003046
1306     003048
1307     003050
1308     003052
1309     003054
1310     003056
1311     003058
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

.TITLE VAC2 - VNP TEMPLATE ACTION ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - TEMPLATE SYNTAX TPARS ACTION ROUTINES, PART 2

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

408 000752

409

410

411

412

413 000754

414 000762

415 000770

416

417

40\$: RETURN

;; ERROR CONDITIONS

101\$: MSG\$R 41

111\$: MSG\$R 42

CTLERR: MSG\$R 22

.DSABL LSB

; 'MPLHD' OPERATOR MISSING

; 'LFLHD' OPERATOR MISSING

; FILE COUNT TOO LARGE

```

56      ;**
57      ; LIBRARY MACROS
58      ;**
59      .MCALL  NTLERS$
60      .ENABL  LC
61
62      ;**
63      ; LOCAL DATA
64      ;**
65
66      000000      .PSECT  DATA,D
67
68      .NLIST  BEX
69
70      ;
71      ; SYMBOL NAME IN ASCII
72      ;
73      000000      ASYM:  .BLKB  6
74
75      ;
76      ; GLOBAL SYMBOL LOOKUP CONTROL BLOCK
77
78      000006      000000  000000      GBLK:  .WORD  0,v
79      000012      GSYM:  .BLKW  2
80      000016      GVAL:  .BLKW  1
81
82      ;
83      ; GLOBAL SYMBOL LOOKUP ERRORS
84      ;
85      000020      000032'      GETAB:  .WORD  $ERR27      ; UNDEFINED SYMBOL (WARNING)
86      000022      000054'      .WORD  $ERR29      ; OPEN FAILURE
87      000024      000114'      .WORD  $ERR30      ; READ FAILURE
88      000026      000154'      .WORD  $ERR31      ; READ OVERFLOW
89      000030      000204'      .WORD  $ERR32      ; BAD STB FILE
90
91      ;
92      ; ERROR MESSAGES
93      ;
94      000032      NTLERS$  ,27,2A,$TWARN,REP2A,$ELINE,<undefined>
95      000054      NTLERS$  ,29,4,$TERR,$PRV.N.,<Open failure ( -***. ).>
96      000114      NTLERS$  ,30,4,$TERR,$PRV.N.,<Read failure ( -***. ).>
97      000154      NTLERS$  ,31,4,$TERR,$REP.P.,<Read overflow.>
98      000204      NTLERS$  ,32,4,$TERR,$REP.P.,<Format error.>
99
100      ;
101      ; ERROR MESSAGE FORMAT STRINGS
102      ;
103      000232      052      040      124      FMT2A:  .ASCIIZ  '* Template -- Symbol * '
104      000262      052      040      123      FMT4:  .ASCIIZ  '* Symbol table file -- '
105      .EVEN
106
107      ;
108      ; SYMBOL TABLE UIC AND EXTENSION
109      ;
110      000312      123      124      102      STBEXT:  .ASCIIZ  'STB'
111      .EVEN
112

```

```

190
191      ;+ $CORE1 - ".CORE" OPERAND 1 ACTION ROUTINE
192
193      ; INPUTS:
194      $VALUE=NUMBER OF BYTES NEEDED
195
196      ; OUTPUTS:
197      $CVAL=NUMBER OF BYTES NEEDED
198      $VALUE=DEFAULT MAPPING DISPLACEMENT
199      $CAPR=APR NUMBER
200      ;-
201      $CORE1::
202
203      000312 005767 000000G   TST     .MGMGE      ; MEMORY MANAGEMENT SYSTEM ?
204      000316 100424          BMI     30$          ; NO
205
206      000320 016700 000000G   MOV     $VALUE,R0    ; GET EXPRESSION VALUE
207      000324 062700 000077   ADD     #77,R0      ; ROUND UP TO NEXT MULTIPLE OF 100
208      000330 006000          ROR     R0           ; ...
209      000332 000241          CLC                ; ...
210      000334 006000          ROR     R0           ; ...
211      000336          ASR$     4,R0              ; ...
212      000346 010067 000000G   MOV     R0,$CVAL     ; SET NEW EXPRESSION VALUE
213      000352 012767 000000G 000000G   MOV     #.BASEB,$VALUE ; SET DEFAULT MAPPING
214      000360 116767 000022' 000000G   MOVB   MAPBYT-1,$CAPR ; SET DEFAULT APR NUMBER
215      000366          RETURN
216
217      000370 016767 000000G 000000G 30$: MOV     $VALUE,$CVAL ; COPY EXPRESSION VALUE
218      000376          RETURN

```


FILEID**VAC5

```

VV      VV      AAAAAA      CCCCCCCC      5555555555
VV      VV      AAAAAA      CCCCCCCC      5555555555
VV      VV      AA      AA      CC      55
VV      VV      AA      AA      CC      55
VV      VV      AA      AA      CC      555555
VV      VV      AA      AA      CC      555555
VV      VV      AA      AA      CC      55
VV      VV      AA      AA      CC      55
VV      VV      AAAAAAAA      CC      55
VV      VV      AAAAAAAA      CC      55
VV      VV      AA      AA      CC      55
VV      VV      AA      AA      CC      55
VV      VV      AA      AA      CC      55
VV      VV      AA      AA      CC      55
VV      VV      AA      AA      CCCCCCCC      555555
VV      VV      AA      AA      CCCCCCCC      555555

```

```

....
....
....
....

```

```

LL      SSSSSSSS      TTTTTTTTTT
LL      SSSSSSSS      TTTTTTTTTT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SSSSSS      TT
LL      SSSSSS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LLLLLLLLLLLL      SSSSSSSS      TT
LLLLLLLLLLLL      SSSSSSSS      TT

```

54
55
56
57
58
59

.SBTTL MACRO DEFINITIONS
;***
; LIBRARY MACROS
;***
.MCALL EMSG\$,NTLER\$,RESRG,SAVRG
.ENABL LC

ASCII = 000001R	002 G\$STPP= 000000	N\$SBUF= 000001	X3BHI = 000013	\$RQ1 = ***** GX
A\$CHK= 000000	G\$STSS= 000000	N\$SLDV= 000001	X3BLO = 000003	\$RQ5 = ***** GX
A\$CPS= 000000	G\$STTK= 000000	N\$SMCP= 000001	X3WHI = 000002	\$RQ7 = ***** GX
A\$SPI= 000000	G\$SWRD= 000000	N\$SMML= 000001	X3WLO = 000001	\$SIZE = ***** GX
A\$TRP= 000000	I\$SRAR= 000000	N\$SMOV= 000010	\$BFPTR= ***** GX	\$TERR = ***** GX
CHECK 000322R	I\$SRDN= 000000	N\$SNCT= 000001	\$CAPR = ***** GX	\$UMROF 000544RG
C\$CKP= 000000	K\$SCNT= 177546	N\$SPEM= 000001	\$CAT5 = ***** GX	\$UMRON 000512RG
C\$ORE= 000400	K\$CSR= 177546	P\$SP45= 000000	\$ELINE= ***** GX	\$VALUE= ***** GX
C\$SRSH= 177564	K\$SLDC= 000000	P\$SWRD= 000000	\$ERRZK 000012R	002 \$VFLLC 001022R
D\$BUG= 177514	K\$STPS= 000074	Q\$SOP1= 000010	\$ERR89 000054R	002 \$VFLV1= ***** GX
D\$ISK= 000000	LB.ADR= ***** GX	R\$SDER= 000000	\$ERR90 000102R	002 \$VXLLC 001042R
D\$LL1= 000001	LB.USE= ***** GX	R\$SK11= 000001	\$ERR91 000136R	002 \$X2CHW 000700RG
D\$SYNC= 000000	LD\$LP = 000000	R\$SSND= 000000	\$FLAGS= ***** GX	\$X2CHW 000554RG
D\$SYNM= 000000	L\$ASG= 000000	R\$S11M= 000000	\$FLAG1= ***** GX	\$X3CHB 000742RG
E\$XPR= 000000	L\$DRV= 000000	S\$SWRG= 000000	\$FLAG2= ***** GX	\$X3CHW 000616RG
FL.CHA= ***** GX	L\$SP11= 000001	S\$YSZ= 0076C0	\$FNR50= ***** GX	.BASEB= ***** GX
FL.DDM= ***** GX	L\$S11R= 000000	T\$KMG= 000000	\$GETLB= ***** GX	.CXLB= ***** GX
FL.DLC= ***** GX	MAPBYT 000000R	002 T\$KMIN= 000000	\$LBADR= ***** GX	.CXLB1= ***** GX
FL.LLC= ***** GX	M\$CRB= 000124	V\$CTR= 001000	\$LBIAS= ***** GX	.LINKS= ***** GX
FMT2 = 000166R	002 M\$CRX= 000300	X\$SDBT= 000000	\$LIBR1 000000RG	.MGEXT= ***** GX
FM.2 = 000000	M\$SCS= 000000	X2BHI = 000003	\$LINKS 000464RG	.MGMGE= ***** GX
F\$LVL= 000001	M\$MGE= 000000	X2BLK = 000002	\$NTLHB= ***** GX	.PSTCN= ***** GX
F1.UMR= ***** GX	M\$NET= 000000	X2BLO = 000001	\$NTLPT= ***** GX	.PSTPT= ***** GX
F2.FIL= ***** GX	M\$OVR= 000000	X2WHI = 000005	\$QSTRT= ***** GX	.X2CH1= ***** GX
F2.LBR= ***** GX	N\$ACC= 000001	X2WLO = 000004	\$REP.C= ***** GX	.X3C01= ***** GX

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
001062 001 (RW,I,LCL,REL,CON)
DATA 000206 002 (RW,D,LCL,REL,CON)
Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 10184 Words (40 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:15.22
SY:VAC6.V2,[132,134]VAC6/CR/-SP=SY:[1,1]RSXCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXCM/PA:1,[132,10]VAC6

```

276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294

```

```

      * $LBERR - CLEANUP AFTER AN ERROR
      INPUTS:
      F2.LBR SET IF LIBRARY WAS ALREADY RESIDENT
      LBLK=LIBRARY BLOCK ADDRESS IF IT WAS ALLOCATED
      -
$LBERR:
      BIT    #F2.LBR,$FLAG2    : WAS LIBRARY RESIDENT?
      BNE    10$               : IF NE, YES
      CALL   $CLDEQ            : CLOSE LIBRARY FILE
      MOV     LBLK,R0           : GET LIBRARY BLOCK ADDRESS
      BEQ     20$               : IF EQ, NONE ALLOCATED
      MOV     #1,R1             : ASSUME MAPPED TARGET
      TST     .MGMGE            : MAPPED TARGET SYSTEM?
      BPL     15$               : IF PL, YES
      MOV     #100,R1           : UNMAPPED ALLOCATION SIZE
      CALL    $DEA18            : DE-ALLOCATE LIBRARY BLOCK
      15$:   CALL
      20$:   RETURN

```

```

000570
000570 032767 000000G 000000G
000576 001002
000600
000604 016700 000000'
000610 001411
000612 012701 000001
000616 005767 000000G
000622 100002
000624 012701 000100
000630
000634

```

```
VV      VV      AAAAAA      CCCCCCCC      999999
VV      VV      AAAAAA      CCCCCCCC      999999
VV      VV      AA          AA      CC      99      99
VV      VV      AA          AA      CC      99      99
VV      VV      AA          AA      CC      99      99
VV      VV      AA          AA      CC      99      99
VV      VV      AA          AA      CC      99      99
VV      VV      AA          AA      CC      99999999
VV      VV      AA          AA      CC      99999999
VV      VV      AAAAA;AAAAA      CC      99
VV      VV      AAAAAAAAAAAA      CC      99
VV      VV      AA          AA      CC      99
VV      VV      AA          AA      CC      99
VV      VV      AA          AA      CC      99
VV      VV      AA          AA      CCCCCCCC      999999
VV      VV      AA          AA      CCCCCCCC      999999
```

```

LL          SSSSSSSS  TTTTTTTTTT
LL          SSSSSSSS  TTTTTTTTTT
          SS          TT
LL          SS          TT
LL          SS          TT
LL          SS          TT
LL          SSSSSS    TT
LL          SSSSSS    TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LLLLLLLLLL SSSSSSSS  TT
LLLLLLLLLL SSSSSSSS  TT

```

```

54      ;**
55      ; LIBRARY MACROS
56      ;**
57      .MCALL  EMSGR$,NTLERS$,GETAD$,PUTAD$,SWTCH1$,SWTCHO$,ASL$,PUTRC$,NKRDf$
58      .MCALL  GETRV$,RESRG$,SAVRG
59
60      .IF DF  R$$MPL ;[MP01]
61      .MCALL  KRBDf$,CTBDF$ ;[MP01]
62      KRBDf$ ; DEFINE KRB OFFSETS (M+) ;[MP01]
63      CTBDF$ ; DEFINE CTB OFFSETS ;[MP01]
64      .ENDC
65
66      .ENABL  LC
67
68      .IF  NDF,R$$MPL
69 000000 NKRDf$ ; DEFINE NETWORK KRB OFFSETS
70      .ENDC
71
72      ;**
73      ; LOCAL DATA
74      ;**
75
76 000000 .PSECT  DATA,D
77
78      ;
79      ; STORAGE FOR PDV INDEX
80
81 000000 PDVX: .BLKW  1
82
83      ;
84      ; INTERRUPT BITS TABLE
85
86 000002 000000G ITAB: .WORD  F1.IN1
87 000004 000000C .WORD  F1.IN1!F1.IN2
88 000006 000000C .WORD  F1.IN1!F1.IN2!F1.IN3
89
90      ;
91      ; CPU NUMBER
92
93 000010 000000 CPU: .WORD  0
94
95      ;
96      ; ERROR MESSAGES
97      ;
98
99 000012 NTLERS$ ,33,1,$TWE,$REP,P,,<Line table empty.>
100 000044 NTLERS$ ,34,1,$TERR,$REP,P,,<Line table allocation failure.>
101
102 000114 NTLERS$ ,35,3,$TERR,$REP,P,,<No interrupt linkage.>
103 000152 NTLERS$ ,36,3,$TERR,$REP,P,,<Incomplete interrupt linkage.>
104 000220 NTLERS$ ,37,3,$TERR,$REP,P,,<".MPTAB" operator missing.>
105 000264 NTLERS$ ,38,3,$TERR,$REP,P,,<".LFILE" operator missing.>
106 000330 NTLERS$ ,86,1,$TERR,$REP,P,,<Unload block allocation failure.>
107 000402 NTLERS$ ,88,3,$TERR,$REP,P,,<CTB allocation failure>
108 000442 NTLERS$ ,90,1,$TERR,$REP,P,,<KRB allocation failure>
109 000502 NTLERS$ ,9D,1,$TERR,$REP,P,,<ICB allocation failure>
110

```

```

637
638
639
640
641
642
643
644
645
646
647
648 001332
649 001332 010446
650 001334 016746 000000G
651 001340 016704 000000G
652 001344 022467 000000G
653 001350 001403
654 001352 005316
655 001354 003373
656 001356 000004
657 001360 005726
658 001362 166704 000000G
659 001366 005744
660 001370 010467 000000'
661 001374 012604
662 001376

;+
; GETIND - GET PDV INDEX
;
; INPUTS:
; $PDVA = PDV ADDRESS OF CURRENT PROCESS
;
; OUTPUTS:
; PDVX = PDV INDEX OF CURRENT PROCESS
;
;-

GETIND:
MOV R4,-(SP) ; SAVE R4
MOV $PDVNM,-(SP) ; GET # OF PDV ENTRIES
MOV $PDVTA,R4 ; GET ADDRESS OF PDV MAPPING TABLE
10$: CMP (R4)+,$PDVA ; IS THIS OUR PDV ?
BEQ 20$ ; IF EQ, YES
DEC (SP) ; ANY MORE PDVS ?
BGT 10$ ; IF GT, YES
IOT ; OOPS - CAN'T FIND PDV
20$: TST (SP)+ ; CLEAN UP STACK
SUB $PDVTA,R4 ; CALCULATE PDV INDEX
TST -(R4) ;
MOV R4,PDVX ; SAVE PDV INDEX
MOV (SP)+,R4 ; RESTORE R4
RETURN

```

```

195      .SBTTL .DEAPR - DEALLOCATE FROM PROCESS
196      .SBTTL .DEACB - DEALLOCATE FROM DSR
197
198      ***.DEACB (.DEAB1)
199
200      CALLED TO RETURN 16-BIT MEMORY TO FREE LISTHEAD. ENTRY
201      AT .DEAPR RETURNS MEMORY TO PROCESS FREE SPACE. ENTRY
202      AT .DEAC1 RETURNS MEMORY TO LISTHEAD PASSED IN R3. ENTRY
203      AT .DEACB RETURNS MEMORY TO SCOM.
204
205      INPUTS:
206      R0      - ADDRESS OF THE CORE BUFFER TO BE DEALLOCATED.
207      R1      - SIZE OF THE CORE BUFFER TO DEALLOCATE IN BYTES.
208      R3      - ADDRESS OF LISTHEAD ( .DEAC1 )
209      $BIAS    - BIAS OF THE PROCESS FOR PARTITION DEALLOCATION
210      $PBIAS   - BIAS OF NETWORK POOL FOR NETWORK POOL DEALLOCATION
211      $PDVA    - PDV OF PROCESS FOR PARTITION DEALLOCATION
212
213      OUTPUTS:
214      THE CORE BLOCK IS MERGED INTO THE FREE CORE CHAIN BY CORE
215      ADDRESS AND IS AGGLOMERATED IF NECESSARY WITH ADJACENT BLOCKS.
216      R3 IS DESTROYED
217
218
219      000212 012703 000000G .DEAPR:MOV    #$PRAVL,R3    ; DEALLOCATE FROM PARTITION
220      000216 016767 000000G 000000G MOV    $BIAS,$RBLCK ; BIAS OF THE PROCESS
221
222      .IF DF R$$MPL ;[MP01]
223
224      .DEAIC: BIS    #DF.PDE,$DFLAG ; SET PROCESS OR ICB DEALLOCATION FLAG ;[MP01]
225      .ENDC ;[MP01]
226
227      000224 000404 BR      .DEAC2 ; DEALLOCATE
228
229      000226 012703 000000G .DEACB:MOV    #$CRAVL,R3    ; ADDRESS OF SCOM LISTHEAD
230
231      000232 005067 000000G .DEAC1::CLR    $RBLCK ; WHEN CLEAR DEALLOCATE FROM POOL
232      000236 012704 000000G .DEAC2::MOV    #$BFR,R4    ; POINTER TO WORK BUFFER
233      000242 010546 MOV     R5,-(SP) ; SAVE R5
234      000244 010305 MOV     R3,R5 ; COPY LISTHEAD ADDRESS
235      000246 012767 000004 000000G MOV     #4,$RSIZE ; SET TRANSFER SIZE
236      000254 005743 TST     -(R3) ; POINT AT ROUNDING FACTOR
237      000256 061301 ADD     (R3),R1 ; ROUND ALLOCATION UP
238      000260 042301 BIC     (R3)+,R1 ; BY ROUNDING FACTOR
239      000262 001462 BEQ     60$ ; ZERO LENGTH ?
240
241      .IF NDF R$$MPL ;[MP01]
242
243      000264 GETAD    (R3) ; READ IN FIRST BLOCK ;[MP01]
244      .IFF ;[MP01]
245
246      BIT     #DF.PDE,$DFLAG ; PROCESS SPACE OR ICB DEALLOCATION ? ;[MP01]
247      BNE     1$ ; IF YES - BRANCH ;[MP01]
248      GETRV   (R3) ; POSSIBLE D-SPACE READ ;[MP01]
249      BR      2$ ; CONTINUE ;[MP01]
250      1$: GETAD    (R3) ; READ FROM PROCESS SPACE ;[MP01]
251

```


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

.TITLE VAL22 - VNP BLOCK ALLOCATION ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - DOUBLEWORD ADDRESSABLE ALLOCATION ROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

```

115
116          ;+ $RLBL - READ LABEL BLOCK
117          ;
118          ; INPUTS:
119          ;   RC=BUFFER ADDRESS
120          ;
121          ; OUTPUTS:
122          ;   C-BIT=SUCCESS/FAILURE
123          ;   $LBN=FIRST TEXT BLOCK LOGICAL BLOCK NUMBER
124          ;   $LLEN=512.
125          ;-
126 000026          $RLBL::
127 000026 012767 001000 000030'  MOV    #512, $LLEN      ; SET BUFFER LENGTH
128 000034          CALL    $READ      ; READ LABEL BLOCK
129 000040 006016      ROR    (SP)      ; SAVE C-BIT
130 000042 066067 000000G 000036'  ADD    L$BHRB(RO), $LBN+2 ; SET LBN OF FIRST TEXT BLOCK
131 000050 005567 000034'      ADC    $LBN      ;
132 000054 006316          ASL    (SP)      ; RESTORE C-BIT
133 000056          RETURN

```

E 15

CREF 04.00

CALL	6-114	6-119			
DIR\$	#6-137	6-137	#6-157	6-157	
GREG\$S	#5-76	6-157			
GTSK\$S	#5-76	6-157			
MOV\$	6-137	6-137	#6-157	6-157	6-157
OFF\$	#6-137	#6-157			
RETURN	6-115	6-174			

```

365                                     .SBTTL $ALN18 - ALLOCATE NON-UMR NETWORK POOL
366                                     .SBTTL $ALL18 - ALLOCATE FROM UMR NETWORK POOL
367
368 *
369 $ALN18 - ALLOCATE NON-UMR MAPPED STORAGE (NETWORK BUFFER POOL)
370 $ALL18 - ALLOCATE DOUBLEWORD ADDRESSABLE STORAGE (NETWORK BUFFER POOL)
371 $ALL22 - ALLOCATE DOUBLEWORD ADDRESSABLE STORAGE (NETWORK BUFFER POOL)
372
373 INPUTS:
374 R1 - BLOCK COUNT (MAPPED) OR BYTE COUNT (UNMAPPED) NEEDED
375 $GBIAS - BIAS OF BASE OF NEXT FREE SEGMENT
376
377 OUTPUTS:
378 C-BIT - SUCCESS/FAILURE
379 RO - BIAS OF BASE OF BUFFER
380 $GBIAS - BIAS OF NEXT FREE SEGMENT
381 ALL OTHER REGISTERS PRESERVED
382
383 NOTE:
384 THIS ROUTINE ALLOCATES ON 32-WORD BOUNDARIES, AND THUS
385 THE BUFFER VIRTUAL ADDRESS LOW SIX BITS WILL ALWAYS BE
386 ZERO. IF THE CALLER INTENDS TO RETURN THE MEMORY ALLOCATED
387 VIA THE $DEA22 ROUTINE THE BUFFER SIZE ( AS CALLED ) MUST
388 BE PRESERVED. THE GLOBAL VALUE $GBIAS MUST BE PROPERLY
389 INITIALIZED BEFORE THIS ROUTINE IS CALLED. SEE COMMENTS
390 IN THE .AMEM ROUTINE ( MODULE 'VAL22' ).
391
392 $ALN18::
393 001170 BIT #FE,CEX,$FMASK ; IS COMEXEC LOADED ?
394 001176 BEQ 101$ ; NO, THEN VNP ALLOCATIONS FAIL
395
396 001200 JSR R5,$SAVRG ; SAVE R3,R4,R5
397 001204 MOV R2,-(SP) ; SAVE R2
398 001206 MOV R1,-(SP) ; SAVE R1
399 001210 MOV R1,R5 ; COPY BLOCK COUNT
400 001212 MOV $NBIAS,R4 ; GET POINTER TO NON-MAPPED AREA
401 001216 CALL .AMEM ; TRY TO ALLOCATE
402 001222 BCS 13$ ; IF CS, FAILURE - TRY MAPPED AREA
403 001224 MOV R4,$NBIAS ; SET NEW POINTER VALUE
404 001230 BR 15$
405
406 $ALL22::
407 001232 $ALL18::
408 001232 BIT #FE,CEX,$FMASK ; IS COMEXEC LOADED ?
409 001240 BEQ 101$ ; NO, THEN VNP ALLOCATIONS FAIL
410
411 001242 JSR R5,$SAVRG ; SAVE R3,R4,R5
412 001246 MOV R2,-(SP) ; SAVE R2
413 001250 MOV R1,-(SP) ; SAVE R1
414 001252 MOV R1,R5 ; COPY BLOCK COUNT
415 001254 MOV $GBIAS,R4 ; LOAD POINTER IN R4
416 001260 CALL .AMEM ; TRY TO ALLOCATE FROM NETWORK POOL
417 001264 BCS 14$ ; IF CS, FAILURE
418 001266 MOV R4,$GBIAS ; UPDATE $GBIAS
419 001272 MOV R5,RO ; RETURN BIAS IN RO
420 001274 MOV (SP)+,R1 ; RESTORE R1
421 001276 MOV (SP)+,R2 ; RESTORE R2

```

```

55      ;**
56      ; LIBRARY MACROS
57      ;**
58      .MCALL ASR$,EMSG$,EMSG$R,NTLER$,GETRV
59      .ENABLE LC
60
61      ;**
62      ; LOCAL DATA
63      ;**
64
65      .PSECT DATA,D
66
67      .NLIST BEX
68
69
70      ;
71      ; TEMPLATE EXTENSION
72      .IMPEXT: .ASCIZ 'DAT'
73
74      .EVEN
75
76      ;
77      ; NONSENSE INTERRUPT TABLE (RSX11M)
78      ;
79
80      NSITAB: .WORD .NS0
81              .WORD .NS1
82              .WORD .NS2
83              .WORD .NS3
84              .WORD .NS4
85              .WORD .NS5
86              .WORD .NS6
87              .WORD .NS7
88
89      ;
90      ; ERROR MESSAGES
91      ;
92      NTLER$ ,17,3,$TERR,$PRV.N,$ELINE,<Open failure ( -***. ).>
93
94      NTLER$ ,23,2,$TERR,$REP.C,$ELINE,<Interrupt linkage error.>
95      NTLER$ ,24,2B,$TERR,REP2B,$ELINE,<not in system.>
96      NTLER$ ,25,2B,$TERR,REP2B,$ELINE,<not available.>
97
98      NTLER$ ,41,2,$TERR,$REP.C,$ELINE,<'MPLHD' operator missing.>
99      NTLER$ ,42,2,$TERR,$REP.C,$ELINE,<'LFLHD' operator missing.>
100
101      NTLER$ ,43,2,$TERR,$REP.C,$ELINE,<Secondary table missing.>
102
103      NTLER$ ,22,2,$TERR,$REP.C,$ELINE,<File count too large.>
104
105      NTLER$ ,81,2,$TERR,$REP.C,$ELINE,<MUX operator>
106      NTLER$ ,85,2,$TERR,$REP.C,$ELINE,<COMIOP operator>
107      NTLER$ ,KX,2,$TERR,$REP.C,$ELINE,<Resource allocation failure.>
108      NTLER$ ,KY,2,$TERR,$REP.C,$ELINE,<Only 14. allocations/libraries allowed.>
109
110      ;
111      ; ERROR MESSAGE FORMAT STRINGS

```

```

419
420
421
422
423
424
425
426
427
428
429
430
431
432 000776
433 000776
434 001002
435 001006 012700 000000G
436 001012
437 001016 016767 000000G 000000G
438 001024 016767 000000G 000000G
439 001032 012700 000000G
440 001036
441 001042 010067 000000G
442 001046
443
444
445 001050
446 001054 012700 000000G
447 001060 022767 000400 000000G 5$:
448 001066 103740
449 001070 005767 000000G
450 001074 001432
451 001076
452 001102 103427
453 001104 110022
454 001106 116712 000000G
455 001112 016746 000000G
456 001116
457 001122
458 001126 012600
459 001130
460 001134 103412
461 001136 112712 000000G
462 001142 166700 000000G
463 001146 000402
464 001150 160067 000000G 10$:
465 001154 005367 000000G 20$:
466 001160 001373
467 001162
468
469

```

```

+
$MXTAB - ".MXTAB" ACTION ROUTINE
$CFILE - ".FILE" ACTION ROUTINE IF COUNT IS SPECIFIED

INPUTS:
$$MUX/$VALUE=FILE REPEAT COUNT
$FILE=SECONDARY TEMPLATE FILE NAME

OUTPUTS:
R0,R1,R2=DESTROYED
-

.ENABL LSB
$MXTAB::
CALL $VFDDM ; MAKE SURE IT'S A DDM
CALL $VFMUX ; AND A MUX LINE
MOV #F1,MXT,R0 ; MAKE SURE THIS IS THE ONLY OCCURANCE
CALL $VFSPC ;
MOV $SIZE,$OFF ; USE CURRENT SIZE AS OFFSET VALUE
MOV $MUX,$VALUE ; DO IT <UNIT COUNT> TIMES
MOV #.MXTAB,R0 ; FUNCTION CODE TO BE USED
CALL 5$ ; PROCESS SECONDARY TEMPLATE
MOV R0,$MXLEN ; SAVE MUX TABLE LENGTH
RETURN

$CFILE::CALL $EVEN ; AUTOMATIC WORD ALIGNMENT ($FILE NOT ALIGNED)
MOV #.FILE,R0 ; FUNCTION CODE TO BE USED
CMP #256,$VALUE ; IS COUNT > 256.?
BLO CTLERR ; IF LO, YES
TST $VALUE ; IS COUNT 0?
BEQ 30$ ; IF EQ, YES
CALL $R02 ; REQUEST BINARY FILE
BCS 30$ ; IF CS, NO ACCESS
MOVB R0,(R2)+ ; STORE FUNCTION CODE
MOVB $VALUE,(R2) ; AND FILE REPEAT COUNT
MOV $SIZE,-(SP) ; SAVE CURRENT LINE TABLE SIZE
CALL $FILE ; PROCESS SECONDARY TEMPLATE
CALL $EVEN ; AUTOMATIC WORD RE-ALIGNMENT
MOV (SP)+,R0 ; RESTORE SIZE FROM START OF FILE
CALL $R01 ; REQUEST BINARY FILE AGAIN
BCS 30$ ; IF CS, NO ACCESS
MOVB #.EOF,(R2) ; STORE END OF FILE FUNCTION CODE
SUB $SIZE,R0 ; GIVING AMOUNT ADDED BY SECONDARY TEMPLATE
BR 20$
10$: SUB R0,$SIZE ; ADJUST DATA BASE SIZE
20$: DEC $VALUE ; BY <FILE COUNT-1> * <FILE SIZE>
BNE 10$ ; ...
30$: RETURN

.DSABL LSB

```

113
114 000000

.PSECT

```

220          ;+
221          ; $CORE2 - ".CORE" OPERAND 2 ACTION ROUTINE
222
223          ; INPUTS:
224          ; $VALUE=MAPPING REGISTER ADDRESS
225
226          ; OUTPUTS:
227          ; RO=DESTROYED
228          ; $VALUE=MAPPING DISPLACEMENT
229          ; $CAPR=APR NUMBER
230          ; -
231
232          $CORE2::
233          000400          TST      .MGMGE          ; MEMORY MANAGEMENT SYSTEM ?
234          000400          BMI      40$          ; NO
235
236          000406          MOV      $VALUE,RO      ; GET EXPRESSION VALUE
237          000412          BEQ      25$          ; IF ZERO, USE DEFAULT
238          000414          BIT      #1,RO          ; IS IT ODD?
239          000420          BNE      20$          ; IF NE, YES
240
241          ; IF NDF R$$MPL ;[CMP01]
242          000422          SUB      #KISARO,RO      ; IS IT WITHIN KERNEL PAGE REGISTER SET?
243          000426          BMI      10$          ; IF MI, NO
244          000430          CMP      #16,RO          ; VALID KERNEL PAGE REGISTER?
245          000434          BHS      30$          ; IF HIS, YES
246          000436          10$: SUB      #UISARO-KISARO,RO ; IS IT WITHIN USER PAGE REGISTER SET?
247          000442          BMI      20$          ; IF MI, NO
248          000444          CMP      #16,RO          ; VALID USER PAGE REGISTER?
249          000450          BHS      30$          ; IF HIS, YES
250
251          ; IFF          ;[CMP01]
252          000452          BR       30$          ;[CMP01]
253
254          .ENDC          ;[CMP01]
255
256          000452          20$: MSG$ 39          ; PRINT WARNING MESSAGE
257          000460          25$: MOV      #-2,RO      ; USE DEFAULT MAPPING
258          000464          30$: MOV      MAPP(RO),$VALUE ; SET MAPPING DISPLACEMENT
259          000472          30$: ASR      RO          ; CONVERT TO BYTE INDEX
260          000474          30$: MOV      MAPBYT(RO),$CAPR ; SET APR NUMBER
261          000502          40$: RETURN

```


.TITLE VAC5 - VNP TEMPLATE ACTION ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - TEMPLATE SYNTAX TPARS ACTION ROUTINES, PART 5

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/R SX V1.0

CONSTANT DEFINITIONS

```

61
62
63          000001          X3WLO = 1          ; LOW VALUE FOR .X3CHW OPERATOR
64          000002          X3WHI = 2          ; HIGH VALUE FOR .X3CHW OPERATOR
65          000003          X3BLO = 3          ; LOW VALUE FOR .X3CHB OPERATOR
66          000013          X3BHI = 11.        ; HIGH VALUE FOR .X3CHB OPERATOR
67          000004          X2WLO = 4          ; LOW VALUE FOR .X2CHW OPERATOR
68          000005          X2WHI = 5          ; HIGH VALUE FOR .X2CHW OPERATOR
69          000001          X2BLO = 1          ; LOW VALUE FOR .X2CHB OPERATOR
70          000003          X2BHI = 3          ; HIGH VALUE FOR .X2CHB OPERATOR
71          000002          X2BLK = 2          ; BLOCK SIZE - .X2CHW SPECIAL CASE (SORRY)

```

VAC6 CREATED BY MACRO ON 29-JUN-85 AT 02:10 PAGE 1 F 7

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
ASC11	000001 R	#7-90 8-128 8-158
CHECK	000322 R	8-170 8-172 8-179
FL.CHA	= ***** GX	9-233
FL.DDM	= ***** GX	16-412
FL.DLC	= ***** GX	16-412
FL.LLC	= ***** GX	15-394
FMT2	= 000166 R	7-97 7-98 7-99 7-100 #7-104
FM.2	= 000000	#7-97 #7-98 #7-99 #7-100
F1.UMR	= ***** GX	9-235 11-279 12-297
F2.FIL	= ***** GX	8-125 8-206
F2.LBR	= ***** GX	8-192 9-241 9-245
LB.ADR	= ***** GX	9-237
LB.USE	= ***** GX	9-239
MAPBYT	= 000000 R	#7-84 8-200
MS\$MGE	= 000000	7-83
RS\$MPL	= *****	8-131
SS\$BAS	= *****	7-97 7-97 7-98 7-98 7-99 7-99 7-100 7-100
X2BHI	= 000003	#6-70 14-362
X2BLK	= 000002	#6-71 13-318 14-364
X2BLO	= 000001	#6-69 14-360
X2WHI	= 000005	#6-68 13-322
X2WLO	= 000004	#6-67 13-320
X3BHI	= 000013	#6-66 14-372
X3BLO	= 000003	#6-65 14-370
X3WHI	= 000002	#6-64 13-330
X3WLO	= 000001	#6-63 13-328
\$BFPTR	= ***** GX	*8-204
\$CAPR	= ***** GX	*8-200
\$CAT5	= ***** GX	8-159 8-161 8-163
\$ELINE	= ***** GX	7-97 7-98 7-99 7-100
\$ERRZK	000012 R	#7-97 11-284
\$ERR89	000054 R	#7-98 15-396
\$ERR90	000102 R	#7-99 16-414
\$ERR91	000136 R	#7-100 13-332 14-374
\$FLAGS	= ***** GX	9-233 15-394 16-412
\$FLAG1	= ***** GX	9-235 *11-279 *12-297
\$FLAG2	= ***** GX	8-125 8-192 *8-206 *9-241 9-245
\$FNR50	= ***** GX	*8-160 *8-162 *8-164 9-225 9-227 9-229
\$GETLB	= ***** GX	8-176
\$LBADR	= ***** GX	*8-165 *9-242 *9-243
\$LBIAS	= ***** GX	*9-222
\$LBRI	000000 RG	#8-124
\$LINKS	000464 RG	#10-259
\$NTLHB	= ***** GX	8-167 8-174 11-282
\$NTLPT	= ***** GX	8-166
\$QSTRT	= ***** GX	9-237
\$REP.C	= ***** GX	7-97 7-98 7-99 7-100
\$RQ1	= ***** GX	10-261 13-336 14-378
\$RQ5	= ***** GX	8-203
\$RQ7	= ***** GX	8-201
\$SIZE	= ***** GX	*10-264 *13-338 *14-380
\$TERR	= ***** GX	7-97 7-98 7-99 7-100

```

296
297
298
299
300
301
302
303
304
305
306
307
308 000636 112314
309 000640 122724 000052
310 000644 005304
311 000646 012700 000000G
312 000652 112024
313 000654 001376
314 000656 005304
315 000660 010500
316 000662 005720
317 000664 122720 000120
318 000670 001375
319 000672 122720 000122
320 000676 001372
321 000700 112710 000060
322 000704 156710 000000G
323 000710
324
325      000001

;+
; REP6 - REPLACE LIBRARY NAME AND THEN FORM APR NUMBER
;
; INPUTS:
;   R3=FORMAT STRING ADDRESS
;   R4=ERROR MESSAGE BUFFER ADDRESS
;   $CAPR=APR NUMBER IN BINARY
;
; OUTPUTS:
;   R0=DESTROYED
;   R3,R4=UPDATED
;
REP6:  MOVB    (R3)+,(R4)      ; COPY FORMAT STRING
        CMPB   #'*,(R4)+      ; LOOKING FOR FIRST ASTERISK
        DEC    R4              ; SKIP THE ASTERISK
        MOV     #$FILE,R0      ; GET ASCIZ FILE NAME ADDRESS
10$:   MOVB    (R0)+,(R4)+      ; COPY THE FILE NAME
        BNE     10$            ;
        DEC     R4              ; SKIP THE ZERO BYTE
        MOV     R5,R0           ; COPY ERROR MESSAGE BLOCK ADDRESS
        TST     (R0)+          ; NOW POINTS AT TEXT STRING
20$:   CMPB    #'P,(R0)+        ; LOOK FOR 'P'
        BNE     20$            ;
        CMPB    #'R,(R0)+        ; FOLLOWED BY 'R'
        BNE     20$            ;
        MOVB    #'0,(R0)        ; FORM APR NUMBER
        BISB    $CAPR,(R0)      ;
        RETURN
;
        .END

```

VAC9 - VNP TEMPLATE ACTION ROUT MACRO V05.03b Saturday 29-Jun-85 02:11 ^{6 9}
Table of contents

7- 193 ALVLIB - ALLOCATE VECTOR BLOCK FOR LIBRARY

```

111
112      : ERROR MESSAGE FORMAT STRINGS
113      :
114      .NLIST BEX
115
116 000542      052      040      000 FMT1: .ASCIZ '* '
117
118 000545      052      040      124 FMT3: .ASCIZ '* Template file -- '
119
120      .LIST BEX
121      .EVEN
122
123 000000      .PSECT

```

```

664
665
666
667
668
669 001400 116723 000000'
670 001404 105023
671 001406 016723 000000G
672 001412 016713 000000G
673 001416 005203
674 001420 005767 000000G
675 001424 001404
676 001426 105243
677 001430 005723
678 001432 016723 000000G
679 001436 010367 000000G
680 001442 116704 000000G
681 001446 006304
682 001450 006304
683 001452 060304
684 001454 010467 000000G
685 001460
686
687
688      000001

;+
; UBLTA - STORE LINE TABLE ADDRESS DATA IN UNLOAD BLOCK
;-
;
UBLTA:  MOVB  PDVX,(R3)+      ; STORE PDV INDEX OF THIS PROCESS
        CLR8  (R3)+          ; CLEAR RESERVED BYTE
        MOV   $$ADDR,(R3)+    ; STORE LINE TABLE ADDRESS
        MOV   $$SIZE,(R3)+    ; AND LINE TABLE SIZE
        INC   R3              ; MAKE IT AN ODD ADDRESS (TEMPORARY)
        TST   $NALL           ; ANY ALLOCS/LIBRS?
        BEQ   10$            ; IF ZERO, NO
        INCB  -(R3)+          ; INDICATE SOME ALLOCS/LIBRS
        TST   (R3)+           ; SKIP LINE TABLE SIZE
        MOV   $NALL,(R3)+     ; STORE ALLOC/LIBR COUNTS
10$:    MOV   R3,$UDA          ; SET UNLOAD DATA VIRTUAL ADDRESS
        MOVB  $NALL,R4        ; GET NUMBER OF ALLOCS
        ASL   R4              ; MULTIPLY BY 4 (2 WORDS PER ENTRY)
        ASL   R4
        ADD   R3,R4           ; ADD UNLOAD BLOCK ADDRESS
        MOV   R4,$LDA         ; GIVING LIBRARY DATA ADDRESS
        RETURN

        .END
  
```

```

252
253
254
255 000274 011314
256 000276 000404
257 000300
258
259 000300
260
261
262
263
264
265
266
267
268
269
270 000310 010302
271 000312 011403
272 000314 001402
273 000316 020003
274 000320 103367
275 000322 010246
276 000324 066416 000002
277 000330 020026
278 000332 001004
279 000334 010200
280 000336 066401 000002
281 000342 000411
282 000344 010014
283 000346 020205
284 000350 001002
285 000352 010015
286 000354 000404
287 000356
288
289
290 000356
291
292
293
294
295
296
297
298
299
300
301 000366 010314
302 000370 010046
303 000372 060116
304 000374 020326
305 000376 001006
306
307
308

2$:
.ENDC

MOV (R3),(R4) ; PREPARE TO ENTER LOOP
BR 15$

10$:
.IF NDF R$$MPL ;[MP01]
GETAD R3 ; READ NEXT FREE BLOCK DESCRIPTOR

.IFF ;[MP01]
BIT #DF.PDE,$DFLAG ; PROCESS SPACE OR ICB DEALLOCATION ;[MP01]
BNE 11$ ; IF YES - BRANCH ;[MP01]
GETRV R3 ; POSSIBLE D-SPACE READ ;[MP01]
BR 12$ ; CONTINUE ;[MP01]
11$: GETAD R3 ; READ FROM PROCESS SPACE ;[MP01]
12$:
.ENDC

15$: MOV R3,R2 ; SAVE ADDRESS OF CURRENT BLOCK
MOV (R4),R3 ; GET ADDRESS OF NEXT BLOCK
BEQ 20$ ; IF EQ END OF LIST
CMP R0,R3 ; BLOCK GO HERE?
BHS 10$ ; IF HIS NO
20$: MOV R2,-(SP) ; CALCULATE HIGHEST ADDR. IN CUR. BLK.
ADD 2(R4),(SP)
CMP R0,(SP)+ ; EQUAL TO BLOCK BEING RELEASED?
BNE 30$ ; IF NE NO
MOV R2,R0 ; REDEFINE ADDR. OF BLOCK BEING RELEASED
ADD 2(R4),R1 ; CALCULATE SIZE OF AGGLOMERATED BLOCK
BR 40$

30$: MOV R0,(R4) ; SET ADDRESS OF NEXT BLOCK
CMP R2,R5 ; IS THIS THE LISTHEAD ?
BNE 35$ ; NO
MOV R0,(R5) ; POINT LISTHEAD AT NODE
BR 40$

35$:
.IF NDF R$$MPL ;[MP01]
PUTAD R2 ; WRITE UPDATE BLOCK DESCRIPTOR ;[MP01]
.IFF ;[MP01]

BIT #DF.PDE,$DFLAG ; PROCESS SPACE OR ICB DEALLOCATION ;[MP01]
BNE 36$ ; IF YES - BRANCH ;[MP01]
PUTRC R2 ; POSSIBLE D-SPACE READ ;[MP01]
BR 37$ ; CONTINUE ;[MP01]
36$: PUTAD R2 ;[MP01]
37$:
.ENDC

40$: MOV R3,(R4) ; ASSUME NO AGGLOMERATION
MOV R0,-(SP) ; CALCULATE HIGHEST ADDRESS IN BLOCK
ADD R1,(SP)
CMP R3,(SP)+ ; EQUAL TO NEXT BLOCK?
BNE 50$ ; IF NE NO

.IF NDF R$$MPL ;[MP01]

```


.....+

THE WORD AT \$BIAS IS THE LISTHEAD FOR A LINKED LIST OF FREE SEGMENTS OF MEMORY. THE CHAIN POINTERS ARE KEPT IN THE FIRST WORD OF EACH SEGMENT AND ARE THE REAL MEMORY ADDRESS MODULO 64 (1/64 REAL MEMORY ADDRESS). THIS IS THE VALUE THAT SHOULD BE PUT INTO THE SCRATCH APR TO MAP TO THE NEXT SEGMENT. THE SECOND WORD OF EACH SEGMENT IS THE SIZE IN 32 WORD BLOCKS OF THE CURRENT SEGMENT. IT IS IMPORTANT TO UNDERSTAND THAT ALL SEGMENTS MUST BE ALLIGNED ON 32 WORD BOUNDARYS, BUT THAT THE LISTHEAD \$BIAS IS NOT SO CONSTRAINED. THIS CAUSES SOME EXCEPTION TESTING IN THESE ROUTINES.

INITIALLY \$BIAS SHOULD CONTAIN THE BIAS (1/64 PHYSICAL MEMORY ADDRESS) FOR THE BASE OF THE MEMORY TO BE AS A BUFFER POOL. THIS POOL CAN BE ALLOCATED IN THE SYSTEM BY EITHER USING A PARTITION OR TASK. ADDITIONALLY THE FIRST WORD OF THE PARTITION/TASK SHOULD BE ZERO (LAST SEGMENT) AND THE SECOND WORD THE SIZE OF THE PARTITION/TASK IN 32-WORD BLOCKS.

INITIALLY:

\$QBIAS -->	0
	SIZE
	REST OF FREE SPACE

AFTER THESE TWO ROUTINES HAVE BEEN RUNNING FOR AWHILE, THE PARTITION/TASK WILL PROBABLY CONTAIN A LINKED LIST OF FREE SEGMENTS OF MEMORY. THE LISTHEAD \$OBIA\$ IS MAINTAINED BY THESE ROUTINES SO THAT IT IS ALWAYS A POINTER TO THE FIRST FREE SEGMENT OF MEMORY. ONE SURE-FIRE WAY TO CRASH THE SYSTEM IS TO MODIFY \$OBIA\$ SINCE THIS ROUTINE CANNOT TELL WHETHER OR NOT IT IS PROCESSING GARBAGE AND CAN ALLOCATE MEMORY FROM ANYWHERE IN THE REAL ADDRESS SPACE.

THESE ROUTINES ALWAYS AGGLOMERATE SEGMENTS WHEN THEY ARE CONTIGUOUS. THESE ROUTINES ARE NOT REENRANT.

NOTE THAT IT IS THE CALLERS' RESPONSIBILITY TO SET UP THE ADDRESS BIAS AND SIZE OF SEGMENTS BEING RETURNED TO THE POOL CORRECTLY. IF EITHER VALUE IS INCORRECT, IT WILL EVENTUALLY CRASH THE SYSTEM.

```

135
136
137
138
139
140
141
142
143
144
145
146
147 000060 $READX:: JSR R2,$SAVVR ; SAVE VOLITAL REGISTERS
148 000060 004267 000000G TST ,MGMGE ; MEMORY MANAGEMNET SYSTEM ?
149
150 000064 005767 000000G BGE 5$ ; YES
151 000070 002006
152
153 000072 006000 ROR R0 ; CONVERT ADDRESS TO BIAS
154 000074 ASR$ 5,R0 ; ...
155
156 000106 016702 000030' 5$: MOV $LLEN,R2 ; R2 = TOTAL NUMBER OF BYTES TO TRANSFER
157 000112 012767 001000 000030' MOV #512,,$LLEN ; TRANSFER THIS MANY BYTES/RECORD
158 000120 010001 MOV R0,R1 ; BIAS OF MEMORY BLOCK
159 000122 012700 000000G MOV #5BFR,R0 ; BUFFER ADDRESS
160 000126 162702 001000 10$: SUB #512,,$R2 ; ATTEMPT TO TRANSFER
161 000132 002002 BGE 20$ ; AT LEAST 512 BYTES.
162 000134 060267 000030' ADD R2,$LLEN ; TRANSFER REMAINDER
163 000140 CALL $READ ; READ IN NEXT BLOCK
164 000144 103424 BCS 30$ ; ERROR DURING READ
165 000146 062767 000001 000036' ADD #1,$LBN+2 ; INCREMENT BLOCK NUMBER
166 000154 005567 000034' ADC $LBN ; DOUBLE WORD
167 000160 PUTAD #0,$LLEN,R1 ; WRITE BLOCK TO VIRTUAL MEMORY
168 000204 062701 000010 ADD #10,R1 ; INCREMENT VIRTUAL BLOCK TO NEXT 512 BYTE BLOCK
169 000210 005702 TST R2 ; ANY BYTES LEFT ?
170 000212 003345 BGT 10$ ; YES
171 000214 000241 CLC ; 'GNAL SUCCESS
172 000216 30$: RETURN

```

```

LL          SSSSSSSS  TTTT TTTT TTTT
LL          SSSSSSSS  TTTT TTTT TTTT
          SS          TT
          SS          TT
          SS          TT
          SS          TT
          SSSSSS      TT
          SSSSSS      TT
          SS          TT
          SS          TT
          SS          TT
          SS          TT
          SS          TT
          SS          TT
          SS          TT
          SSSSSSSS  TT
          SSSSSSSS  TT
LLLLLLLLLLLL SSSSSSSS
LLLLLLLLLLLL SSSSSSSS

```

VCEAL - VNP ALLOCATION/DEALLOCA MACRO V05.03b Saturday 29-Jun-85 02:13 Page 12-1
\$ALL18 - ALLOCATE FROM UMR NETWORK POOL

```
422 001300                                RETURN
423                                     ;
424                                     ; ERROR CONDITIONS
425                                     ;
426 001302 000261                         101$: SEC          ; INDICATE ERROR
427 001304                                RETURN          ; AND LEAVE
```

```
112 000622      052      040      124 FMT3: .ASCIZ '* Template file -- '  
113 000646      052      040      124 FMT2: .ASCIZ '* Template -- '  
114  
115 000665      052      040      124 FMT2B: .ASCIZ '* Template -- Vector * '  
116  
117 .EVEN  
118  
119 000000 .PSECT
```

```

471                                     ;+
472                                     ; $SECSR - ".SECSR" ACTION ROUTINE
473                                     ;
474                                     ; INPUTS:
475                                     ;     NONE
476                                     ;
477                                     ; OUTPUTS:
478                                     ;     R2=DESTROYED
479                                     ; -
480
481 001164                               $SECSR::
482 001164                               CALL    $VFDDM           ; MAKE SURE IT'S A DDM
483 001170                               CALL    $VFMUX           ; AND THAT IT'S A MUX LINE
484 001174 032767 000000G 000000G       BIT     #FL.LMC,$FLAGS ; IS IT A COMIOP DEVICE?
485 001202 001411                               BEQ     101$       ; IF EQ, NO
486 001204                               CALL    $RQ1            ; REQUEST BINARY FILE
487 001210 103405                               BCS     10$       ; IF CS, NO ACCESS
488 001212 112712 000000G               MOVB    #.SECSR,(R2)    ; STORE FUNCTION CODE
489 001216 062767 000002 000000G       ADD     #2,$SIZE        ; ADJUST DATA BASE SIZE
490 001224                               10$:   RETURN
491
492                                     ;
493                                     ; ERROR CONDITION
494
495 001226                               101$:   MSG$R 85          ; NOT COMIOP DEVICE
496

```

```

116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172

```

```

;+
$GLOBL - GLOBAL SYMBOL ACTION ROUTINE
:
: INPUTS:
: .PSTPT=SYMBOL ADDRESS
: .PSTCN=SYMBOL LENGTH
:
: OUTPUTS:
: R0,R1,R2=DESTROYED
: $VALUE=EXTERNAL SYMBOL VALUE
:
$GLOBL:
CLR $VALUE ; SET NO VALUE YET
BIT #F1.STE,$FLAG1 ; ANY SYMBOL TABLE ERRORS YET?
BNE 50$ ; IF NE, YES
MOV .PSTPT,R0 ; GET SYMBOL ADDRESS
MOV #ASYM,R1 ; GET LOCAL SCRATCH PAD ADDRESS
MOV .PSTCN,R2 ; GET SYMBOL LENGTH
CMP #6,R2 ; 6 CHARACTERS OR LESS?
BHS 10$ ; IF HS, YES
MOV #6,R2 ; USE ONLY FIRST 6 CHARACTERS OF SYMBOL
MOV R2,-(SP) ; SAVE LENGTH
MOVB (R0)+,(R1)+ ; COPY SYMBOL
DEC R2
BNE 20$
MOV #6,R2 ; IS SYMBOL LENGTH 6?
SUB (SP)+,R2
BEQ 40$ ; IF EQ, YES
MOVB #'',(R1)+ ; FILL WITH BLANKS
DEC R2
BNE 30$
MOV #ASYM,R0 ; UP TO A LENGTH OF 6
MOV PC,R1 ; CONVERT 6 CHARACTERS
CALL $CAT5 ; FROM ASCII TO RAD50
MOV R1,GSYM ; AND STORE IN CONTROL BLOCK
MOV PC,R1
CALL $CAT5
MOV R1,GSYM+2
CLR R0 ; SETUP ASCII FILE SPEC
CLR R1 ; NOTE - [1,1]XXX.STB FOR 11D/IAS
MOV #STBEXT,R2
CALL $ZFS
MOV #GBLK,R0 ; POINT TO SYMBOL CONTROL BLOCK
CALL $SYMBL ; LOOK FOR GLOBAL SYMBOL
BCS 101$ ; IF CS, FILE ERROR OF SOME SORT
MOV $VAL,$VALUE ; SET SYMBOL VALUE
TST R1 ; CHECK COUNT OF UNDEFINEDS
BNE 111$ ; IF NE, SYMBOL WAS UNDEFINED
50$: RETURN

:
: ERROR CONDITIONS
:
101$: CMPB $CLEVL,#6 ; IS GCL AT MAXIMUM DEPTH?
BEQ 111$ ; IF EQ, YES
BIS #F1.STE,$FLAG1 ; INDICATE SYMBOL TABLE ERROR
MOV GETAB-2(R0),R0 ; GET ERROR MESSAGE BLOCK ADDRESS
111$:

```

```

263      ;+
264      $CORE - ".CORE" ACTION ROUTINE
265
266      : INPUTS:
267      $CVAL=NUMBER OF BLOCKS (MAPPED) OR BYTES (UNMAPPED) NEEDED
268      $VALUE=MAPPING DISPLACEMENT TO BE USED
269
270      : OUTPUTS:
271      C-BIT=SUCCESS/FAILURE
272      R0,R1,R2=DESTROYED
273      $NALL INCREMENTED IF ALLOCATION SUCCEEDED
274      F1.ERR SET IF ALLOCATION FAILED
275      :-
276
277      $CORK:: CALL      $CORE2          ; PROCESS OPERAND 2
278      000504
279
280      $CORE:: CALL      $VFLV1          ; MAKE SURE IT'S THE PRIMARY TEMPLATE
281      000510
282
283      000514 005767 000000G          TST      ,MGMGE          ; MEMORY MANAGEMENT SYSTEM ?
284      000520 100403                    BMI      5$           ; NO
285
286      000522                    CALL     $RQ7          ; REQUEST BINARY FILE
287      000526 000402                    BR       8$
288
289      000530                    5$: CALL     $RQ5          ; DITTO
290
291      000534 103465                    8$: BCS      20$          ; IF CS, NO ACCESS
292      000536 005000                    CLR      R0           ; ASSUME ZERO LENGTH
293      000540 032767 000000G 000000G    BIT      #FL,CHA,$FLAGS ; MUX CHARACTERISTICS UPDATE ONLY ?
294      000546 001402                    BEQ      80$          ; NO ..
295      000550 005067 000000G            CLR      $CVAL        ; YES .. NO-OP THE ALLOCATION
296      000554 016701 000000G            MOV      $CVAL,R1     ; GET BLOCK COUNT
297      000560 001421                    BEQ      10$          ; IF ZERO, DON'T ALLOCATE
298      000562                    CALL     MAX14          ; ALLOW ONLY 14. ALLOCATIONS
299      000566 103450                    BCS      20$          ; IF CS, NO MORE ALLOWED
300      000570 032767 000000G 000000G    BIT      #F1,UMR,$FLAG1 ; IS UMR MAPPING NEEDED ?
301      000576 001003                    BNE      81$          ; IF NE, YES
302      000600                    CALL     $ALN18         ; ALLOCATE FROM NON-MAPPED AREA
303      000604 000402                    BR       82$
304      000606                    81$: CALL     $ALL18        ; ALLOCATE 18 BIT ADDRESSABLE STORAGE
305      000612 103437                    82$: BCS      ALLERR    ; IF CS, FAILURE
306      000614                    CALL     ZER18          ; ZERO OUT 18-BIT ALLOCATION
307
308      000620 105267 000000G            9$: INCB      $NALL      ; TALLY SUCCESSFUL ALLOCATION
309      000624 112722 000000C            10$: MOVB     #.CORE:200,(R2)+ ; STORE FUNCTION CODE
310      000630 110022                    MOVB     R0,(R2)+      ; STORE FIRST WORD OF ADDRESS
311      000632 000300                    SWAB      R0           ;
312      000634 110022                    MOVB     R0,(R2)+      ;
313
314      000636 005767 000000G            TST      ,MGMGE          ; MEMORY MANAGEMENT SYSTEM ?
315      000642 100404                    BMI      11$          ; NO
316
317      000644 116722 000000G            MOVB      $VALUE,(R2)+ ; STORE SECOND WORD OF ADDRESS
318      000650 116722 000001G            MOVB      $VALUE+1,(R2)+ ;
319

```



```

56      ***
57      ; LIBRARY MACROS
58      ***
59      .MCALL  ASL$,EMSG$,EMSG$R,NTLERS$
60      .ENABL  LC
61
62      ***
63      ; LOCAL DATA
64      ***
65
66      000000      .PSECT  DATA,D
67
68      ;
69      ; ERROR MESSAGES
70      ;
71      NTLERS$ ,69,6,$TERR,$PRV.N,$ELINE,<Open failure ( -***. ).>
72      NTLERS$ ,71,6,$TERR,$REP.P,$ELINE,<not contiguous.>
73      NTLERS$ ,72,6,$TERR,$CUR.N,$ELINE,<Label block read failure ( -***. ).>
74      NTLERS$ ,73,6,$TERR,$REP.C,$ELINE,<not a valid binary image.>
75      NTLERS$ ,74,6,$TERR,$REP.C,$ELINE,<size exceeds 32k-32 word blocks.>
76      NTLERS$ ,75,6,$TERR,$REP.C,$ELINE,<Extension size exceeds 32k words.>
77
78      ;
79      ; ERROR MESSAGE FORMAT STRING
80      ;
81      000332      052      040      111  FMT6:$.ASCIZ      '* Image file -- '
82      000335      155      141      147
83      000340      145      040      146
84      000343      151      154      145
85      000346      040      055      055
86      000351      040      000
87
88      .EVEN
89
90      ;
91      ; DEFAULT APR NUMBER
92      ;
93      MAPBYT: .BYTE  <<.BASEB/2>&77777>/10000
94
95      .EVEN
96
97      .PSECT  ..BUF
98
99      ;
100     ; DISK BUFFER FOR LABEL BLOCK READ
101     ;
102     DSKBUF: .BLKW  1
103
104     .PSECT

```

LOCAL DATA

```

73                                     .SBTTL  LOCAL DATA
74                                     ;***
75                                     ; LOCAL DATA
76                                     ;***
77
78 000000                             .PSECT  DATA,D
79
80                                     ;
81                                     ; DEFAULT APR NUMBER
82                                     ;
83                                     .IF DF  M$$MGE
84 000000      000C      MAPBYT: .BYTE  <<.BASEB/2>&77777>/10000
85                                     .ENDC
86
87                                     ;
88                                     ; FILE NAME IN ASCII BLANK FILLED TO NINE CHARACTERS
89                                     ;
90 000001      ASCII:  .BLKB  9.
91
92                                     .EVEN
93
94                                     ;
95                                     ; ERROR MESSAGE
96                                     ;
97 000012      NTLR$ ,ZK,2,$TERR,$REP.C,$ELINE,<UMR's have been disabled.>
98 000054      NTLR$ ,89,2,$TERR,$REP.C,$ELINE,<LLC operator.>
99 000102      NTLR$ ,90,2,$TERR,$REP.C,$ELINE,<DDM or DLC Operator>
100 000136      NTLR$ ,91,2,$TERR,$REP.C,$ELINE,<Illegal operand>
101
102                                     ;
103                                     ; ERROR MESSAGE FORMAT STRING
104 000166      052      040      124  FMT2:  .ASCIZ  '* Template -- '
105      000171      145      155      160
106      000174      154      141      164
107      000177      145      040      055
108      000202      055      040      000
109
110                                     .EVEN
111                                     .PSECT

```

LOCAL DATA

VAC6 CREATED BY MACRO ON 29-JUN-85 AT 02:10 PAGE 2 G 7

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
\$UMROF	000544 RG	#12-296
\$UMRON	000512 RG	#11-278
\$VALUE	= ***** GX	*8-199 13-318 13-320 13-322 13-328 13-330 13-334 14-360 14-362
		14-364 14-370 14-372 14-376
\$VFLLC	001022 R	10-260 13-333 14-375 #15-393
\$VFLV1	= ***** GX	8-196
\$VXLLC	001042 R	13-324 14-366 #16-412
\$X2CHB	000700 RG	#14-359
\$X2CHW	000554 RG	#13-317
\$X3CHB	000742 RG	#14-369
\$X3CHW	000616 RG	#13-327
.BASEB	= ***** GX	7-84 8-199
.CXLBR	= ***** GX	8-174
.CXLBT	= ***** GX	8-169
.LINKS	= ***** GX	10-263
.MGEXT	= ***** GX	11-280
.MGME	= ***** GX	8-197 9-231
.PSTCN	= ***** GX	8-129 8-153
.PSTPT	= ***** GX	8-127
.X2CH1	= ***** GX	13-317 14-359
.X3C01	= ***** GX	13-327 14-369

A\$\$CHK= 000000	G\$\$TTP= 000000	M\$\$FCS= 000000	T\$\$KMG= 000000	\$FLAGS= ***** GX
A\$\$CPS= 000000	G\$\$TSS= 000000	M\$\$MGE= 000000	T\$\$MIN= 000000	\$FLAG1= ***** GX
A\$\$PRI= 000000	G\$\$TTK= 000000	M\$\$NET= 000000	USE= 000444R	\$FLAG2= ***** GX
A\$\$TRP= 000000	G\$\$WRD= 000000	M\$\$OVR= 000000	V\$\$CTR= 001000	\$GETLB= ***** GX
BASE= 000402R	I\$\$RAR= 000000	N\$\$ACC= 000001	XFER= 000120R	\$LBADR= ***** GX
C\$\$CKP= 000000	I\$\$RDN= 000000	N\$\$BUF= 000001	X\$\$DBT= 000000	\$LBERR= 000570RG
C\$\$ORE= 000400	K\$\$CNT= 177546	N\$\$LDV= 000001	\$ALB= ***** GX	\$LBIAS= ***** GX
C\$\$RSH= 177564	K\$\$CSR= 177546	N\$\$MCP= 000001	\$ALL18= ***** GX	\$LBXFR= ***** GX
D\$\$BUG= 177514	K\$\$LDC= 000000	N\$\$MLL= 000001	\$ALN18= ***** GX	\$LIBR= 000000RG
D\$\$ISK= 000000	K\$\$TPS= 000074	N\$\$MOV= 000010	\$BFALL= ***** GX	\$NALL= ***** GX
D\$\$L11= 000001	LBLK= 000000RG	N\$\$NCT= 000001	\$BFBAS= ***** GX	\$NLIB= ***** GX
D\$\$YNC= 000000	LB.APR= ***** GX	N\$\$PEM= 000001	\$BFPTR= ***** GX	\$NTLHB= ***** GX
D\$\$YNM= 000000	LB.USE= ***** GX	P\$\$P45= 000000	\$CAPR= ***** GX	\$PUTLB= ***** GX
E\$\$XPR= 000000	LD\$LP= 000000	P\$\$WRD= 000000	\$CLDEQ= ***** GX	\$REP.C= ***** GX
FL.CHA= ***** GX	LINK= 000340R	Q\$\$OPT= 000010	\$DEA18= ***** GX	\$REP.P= ***** GX
FM12= 000317RG	L\$\$ASG= 000000	REP6= 000636R	\$ELINE= ***** GX	\$SIZE= ***** GX
FM16= 000276RG	L\$\$DRV= 000000	R\$\$DER= 000000	\$ERRY2= 000002R	\$TERR= ***** GX
FM.2= 000000	L\$\$P11= 000001	R\$\$K11= 000001	\$ERRY3= 000076R	\$TWARN= ***** GX
FM.6= 000000	L\$\$11R= 000000	R\$\$SND= 000000	\$ERRY4= 000152R	\$VALUE= ***** GX
F\$\$LVL= 000001	MAX14= 000304R	R\$\$11M= 000000	\$ERRY6= 000220R	\$CXLBR= ***** GX
F1.UMR= ***** GX	M\$\$CRB= 000124	S\$\$WRG= 000000	\$ERRY7= 000036R	\$LIBR= ***** GX
F2.FIL= ***** GX	M\$\$CRX= 000000	S\$\$YSZ= 007600	\$FILEN= ***** GX	\$MGME= ***** GX
F2.LBR= ***** GX				

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
000712 001 (RW,I,LCL,REL,CON)
DATA 000336 002 (RW,D,LCL,REL,CON)
Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 10239 Words (40 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:13.16
SY:VAC7.V2,[132,134]VAC7/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VAC7

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

.TITLE VAC9 - VNP TEMPLATE ACTION ROUTINES
.IDENT /V05.00/

.. COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
.. DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

.. THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
.. ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
.. INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
.. COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
.. OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
.. TRANSFERRED.

.. THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
.. AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
.. CORPORATION.

.. DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
.. SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

.. MODULE DESCRIPTION:

.. VNP - TEMPLATE SYNTAX TPARS ACTION ROUTINES, PART 9
.. DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

.. IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

```

125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148 000000
149 000000 005767 000000G
150 000004 100522
151 000006 032767 000000G 000000G
152 000014 001116
153 000016 032767 000000G 000000G
154 000024 001411
155
156 000026 016701 000000G
157 000032 032761 000000G 000000G
158 000040 001003
159 000042
160 000046 103501
161
162 000050 005067 000000G
163 000054 016701 000000G
164 000060 001475
165 000062 062701 000003
166 000066 042701 000003
167 000072 010167 000000G
168 000076 032767 000000C 000000G
169 000104 001437
170 000106
171 000112 103434
172 000114 005067 000000G
173
174 000120 032767 000000G 000000G
175 000126 001433
176 000130 016701 000000G
177 000134 032761 000000G 000000G
178 000142 001025
179 000144 010002
180 000146 016700 000000G
181 000152 016701 000000G

+
$TMALT - ALLOCATE LINE TABLE

THIS ROUTINE IS CALLED TO ALLOCATE THE PROCESS LINE TABLE. AN ATTEMPT
IS MADE TO FIRST ALLOCATE FROM EXTENSIBLE SPACE WITHIN THE PROCESS. IF
THIS IS SUCCESSFUL AND THE IMAGE IS A MEMORY MANAGEMENT SYSTEM AND WE
ARE ALLOCATING FOR A DDM, THEN A SECONDARY LINE TABLE IS ALLOCATED FROM
DYNAMIC MEMORY. (THE VECTOR FOR THE DDM WILL POINT TO THE SECONDARY
LINE TABLE). THE ADDRESS FOR THE ACTUAL LINE TABLE IS SAVED FOR THE
BINARY EXPANSION ROUTINES.

INPUTS:
$SIZE=NUMBER OF BYTES NEEDED
$NVECT=NUMBER OF VECTORS (FOR VECTOR BLOCK STORAGE ALLOC)

OUTPUTS:
R0,R1,R2=DESTROYED
$ADDR=LINE TABLE ADDRESS
$SADR,$TADR=REAL LINE TABLE ADDRESS
$ICBAD=ICB ADDRESS OR LINE TABLE ADDRESS
$ICB=AND ITS LENGTH
$LINK=LINE TABLE LINK VALUE

$TMALT::
TST $FLAG1 ; ANY ERROR YET?
BMI 100$ ; IF M1, YES - DO NOTHING
BIT #FL,CHA $FLAGS ; MUX CHARACTERISTICS UPDATE ONLY ?
BNE 100$ ; ES.. DON'T ALLOCATE
BIT #FL,DDM LACS ; IS THIS A DDM ?
BEQ 100$ ; IF EQ, NO

MOV $PDVA,R1 ; GET PDV ADDRESS
BIT #ZF,PSE,Z.FLG(R1) ; IS THIS A PSEUDO-DEVICE?
BNE 100$ ; IF YES
CALL KRBAL ; LOCATE A KRB
BCS 100$ ; CS, FAILURE

10$: CLR $LINK ; SET LINE TABLE LINK VALUE
MOV $SIZE,R1 ; GET BYTE COUNT NEEDED
BEQ 101$ ; IF ZERO, ERROR
ADD #3,R1 ; ROUND UP NEXT MULTIPLE OF 4
BIC #3,R1
MOV R1,$SIZE ; RESET LINE TABLE SIZE
BIT #FL,DDM,FL.DLC,$FLAGS ; IS THIS A DDM OR DLC?
BEQ 30$ ; BR IF NO - MUST BE AN LLC
CALL $ALLPR ; ATTEMPT ALLOCATION FROM PROCESS
BCS 30$ ; NO IF C-SET
CLR $ICBAD ; POTENTIAL ICB ADDRESS

BIT #FL,DDM,$FLAGS ; IS THIS A DDM?
BEQ 40$ ; IF EQ NO, SO DO NOTHING
MOV $PDVA,R1 ; GET PDV ADDRESS
BIT #ZF,PSE,Z.FLG(R1) ; IS THIS A PSEUDO-DEVICE?
BNE 40$ ; BR IF YES
MOV R0,R2 ; LINE TABLE ADDRESS SAVED IN R2
MOV $ICB,R0 ; SIZE OF EACH ICB
MOV $NVECT,R1 ; NUMBER OF VECTORS

```

ASSCHK= 000000	F1.LFL= ***** GX	M\$MGE= 000000	\$ADDR = ***** GX	\$NLIB = ***** GX
ASSCP= 000000	F1.MPL= ***** GX	M\$NET= 000000	\$ADRX = ***** GX	\$NTLHB= ***** GX
ASSPRI= 000000	F1.MPT= ***** GX	M\$OVR= 000000	\$ADRZ = ***** GX	\$NTLUB= ***** GX
ASSTRP= 000000	GETIND 001332R	N\$ACC= 000001	\$ALLPR= ***** GX	\$NVECT= ***** GX
CPU 000010R	002 G\$STPP= 000000	N\$BUF= 000001	\$ALL15= ***** GX	\$PDVA = ***** GX
C\$CKP= 000000	G\$STSS= 000000	N\$LDV= 000001	\$ALL16= ***** GX	\$PDVNM= ***** GX
C\$ORE= 000400	G\$STTK= 000000	N\$MCP= 000001	\$ALN18= ***** GX	\$PDVTA= ***** GX
C\$SRSH= 177564	G\$SWRD= 000000	N\$SML= 000001	\$BFR = ***** GX	\$PUTAD= ***** GX
D\$BUG= 177514	ITAB 000002R	002 N\$SMOV= 000010	\$DEA16= ***** GX	\$PUTRC= ***** GX
D\$ISK= 000000	I\$RAR= 000000	N\$NCT= 000001	\$ERR33 000012R	002 \$RADDR= ***** GX
D\$SL11= 000001	I\$RDN= 000000	N\$PEM= 000001	\$ERR34 000044R	002 \$REP.P= ***** GX
D\$SYNC= 000000	KRBAL 000322R	PDVX 000000R	002 \$ERR35 000114R	002 \$RSIZE= ***** GX
D\$YNM= 000000	K\$CNT= 177546	P\$P45= 000000	\$ERR36 000152R	002 \$SADDR= ***** GX
E\$XPR= 000000	K\$CSR= 177546	P\$SWRD= 000000	\$ERR37 000220R	002 \$SIZE = ***** GX
FL.CHA= ***** GX	K\$SDC= 000000	Q\$OPT= 000010	\$ERR84 000264R	002 \$SLTA = ***** GX
FL.DDM= ***** GX	K\$TPS= 000074	R\$DER= 000000	\$ERR86 000330R	002 \$SLTMA= ***** GX
FL.DLC= ***** GX	K.CSR 000002	R\$SK11= 000001	\$ERR88 000402R	002 \$SLTNM= ***** GX
FL.LLC= ***** GX	K.PR1 000000	R\$SND= 000000	\$ERR9C 000442R	002 \$TADDR= ***** GX
FMT1 000542R	002 K.VCT 000001	R\$S11M= 000000	\$ERR9D 000502R	002 \$TERR = ***** GX
FMT3 000545R	LD\$LP = 000000	SYSFDB= ***** GX	\$FLAGS= ***** GX	\$TMALT 000000R6
FM.1 = 000000	L\$ASG= 000000	S\$WRG= 000000	\$FLAG1= ***** GX	\$TMCHK 000514RG
FM.3 = 000000	L\$DRV= 000000	S\$YSZ= 007600	\$GETAD= ***** GX	\$TMPBF= ***** GX
F\$SLVL= 000001	L\$P11= 000001	T\$KMG= 000000	\$ICBAD= ***** GX	\$TWE = ***** GX
F.NRBD= ***** GX	L\$S11R= 000000	T\$SMIN= 000000	\$ICBLN= ***** GX	\$UBIAS= ***** GX
F.RSIZ= ***** GX	L.CTL = ***** GX	UBA 000662R	\$ICBSZ= ***** GX	\$UDA = ***** GX
F.URBD= ***** GX	L.DDM = ***** GX	UBLTA 001400R	\$KRBLN= ***** GX	\$UNBSZ= ***** GX
F1.INT= ***** GX	L.KRBA= ***** GX	V\$CTR= 001000	\$LDA = ***** GX	\$SCSR = ***** GX
F1.IN1= ***** GX	M\$CRB= 000124	X\$SDBT= 000000	\$LINK = ***** GX	\$SPRI = ***** GX
F1.IN2= ***** GX	M\$CRX= 000000	ZF.PSE= ***** GX	\$MUL = ***** GX	\$SVECT= ***** GX
F1.IN3= ***** GX	M\$FCS= 000000	Z.FLG = ***** GX	\$NALL = ***** GX	\$CXUNL= ***** GX
F1.LFI= ***** GX				

. ABS. 000004 000 (RW,I,GBL,ABS,OVR)
 DATA 001462 001 (RW,I,LCL,REL,CON)
 DATA 000572 002 (RW,D,LCL,REL,CON)
 Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 10792 Words (43 Pages)
 Size of core pool: 14440 Words (55 Pages)
 Operating system: RSX-11M/PLUS

Elapsed time: 00:00:23.20
 DB2:VALT.134,[132,134]VALT/CR/-SP=DB2:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VALT

```

309 000400          GETAD R3          ; READ NEXT BLOCK DESCRIPTOR
310          .IFF                                     ;[MP01]
311
312          BIT      #DF.PDE,$DFLAG ; PROCESS SPACE OR ICB DEALLOCATION ? ;[MP01]
313          BNE      46$             ; IF YES - BRANCH ;[MP01]
314          GETRV    R3              ; POSSIBLE D-SPACE READ ;[MP01]
315          BR       47$             ; CONTINUE ;[MP01]
316          46$:     GETAD R3        ; PROCESS SPACE READ ;[MP01]
317          47$:                                     ;[MP01]
318          .ENDC                                     ;[MP01]
319
320 000410 066401 000002      ADD      2(R4),R1 ; CALCULATE SIZE OF AGGLOMERATED BLOCK
321 000414 010167 000002G    50$:     MOV      R1,$BFR+2 ; SET SIZE OF BLOCK BEING RELEASED
322
323          .IF NDF R$$MPL
324 000420          PUTAD R0          ; WRITE UPDATED BLOCK DESCRIPTOR ;[MP01]
325          .IFF                                     ;[MP01]
326
327          BIT      #DF.PDE,$DFLAG ; PROCESS SPACE OR ICB POOL READ ? ;[MP01]
328          BNE      56$             ; IF YES - BRANCH ;[MP01]
329          PUTRC    R0              ; POSSIBLE D-SPACE READ ;[MP01]
330          BR       57$             ; CONTINUE ;[MP01]
331          56$:     PUTAD R0        ; READ FROM PROCESS SPACE ;[MP01]
332          57$:                                     ;[MP01]
333          .ENDC
334
335 000430 012605      60$:     MOV      (SP)+,R5 ; RESTORE R5
336
337          .IF DF R$$MPL
338          BIC      #DF.PDE,$DFLAG ; CLEAR PROCESS SPACE FLAG ;[MP01]
339          .ENDC                                     ;[MP01]
340
341 000432          RETURN
342
343

```


111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167

.SBTTL .AMEM - ALLOCATE VIRTUAL MEMORY IN NTPool

.*
 .AMEM - SUBROUTINE TO ALLOCATE A CONTIGUOUS BLOCK OF MEMORY
 FROM PARTITION.

INPUTS:

R4 - CURRENT VALUE OF \$GBIAS
 R5 - NUMBER OF 32 WORD BLOCKS REQUIRED

OUTPUTS:

C BIT SET IF NO MEMORY ALLOCATED
 CLEAR IF MEMORY ALLOCATED
 R4 - NEW VALUE OF \$GBIAS
 R5 - BIAS OF MEMORY ALLOCATED, IF ALLOCATION SUCCEEDS
 R0,R1,R2 ARE DESTROYED

NOTE:

THE VALUE OF \$GBIAS RETURNED IN R4 MUST BE UPDATED IN
 IT'S GLOBAL STORAGE LOCATION.

REGISTER USAGE:

R0 - BUFFER FOR ADJUSTED HOLE SIZE
 R1 - BIAS OF PREVIOUS HOLE
 R4 - BUFFER FOR BIAS OF NEXT HOLE
 R5 - SIZE (IN 32-WD BLKS) OF SPACE REQUIRED

.MCALL GETAD,PUTAD,ERMSG\$,ERBLK\$,ERRPT\$,ASL\$
 .ENABL LC

.AMEM:: MOV R4,\$RBLCK ; MAP TO LISTHEAD HOLE
 SEC ; ASSUME NO FIRST HOLE
 BEQ 70\$; NO FIRST HOLE
 GETAD #0,#4 ; READ IN LISTHEAD DESCRIPTOR
 MOV \$BFR+2,R0 ; LOAD R0 WITH THIS HOLE SIZE
 SUB R5,R0 ; SUBTRACT DESIRED HOLE SIZE
 BLO 40\$; HOLE TOO SMALL
 BEQ 30\$; HOLE JUST RIGHT
 MOV \$RBLCK,-(SP) ; PUSH CURRENT BIAS ON STACK
 ADD R0,(SP) ; ALLOCATE FROM TOP OF FREE SPACE
 MOV R0,\$BFR+2 ; BASE OF ADJUSTED (SHORTENED) HOLE.
 BR 60\$; RETURN

; DELETE HOLE FROM LIST

30\$: MOV \$RBLCK,-(SP) ; SET BIAS OF ALLOCATED SPACE ON STACK
 MOV \$BFR,R4 ; NEXT HOLE POINTER IS NEW \$GBIAS VALUE
 BR 60\$; RETURN

; SPACE NOT FOUND IN FIRST HOLE -- CHECK REMAINING HOLES (IF ANY)

```

174
175      ;+ $READ - READ A LOGICAL BLOCK
176      ;
177      ; INPUTS:
178      ;   RO=BUFFER ADDRESS
179      ;   $LLEN=BUFFER LENGTH
180      ;   $LBN=LOGICAL BLOCK NUMBER
181      ;
182      ; OUTPUTS:
183      ;   C-BIT=SUCCESS/FAILURE
184      ;
185      $READ::
186      MOV     RO,$LBUF      ; SET BUFFER ADDRESS
187      DIR$    #LBN$DPB     ; READ LOGICAL BLOCK
188      BCS     10$          ; IF CS, DPB ERROR
189      TSTB    $IOSB        ; WAS I/O SUCCESSFUL? (CLEARS C-BIT)
190      BPL     10$          ; IF PL, YES
191      SEC     10$          ; INDICATE FAILURE
192      10$:    RETURN
193
194      000220 010067 000026'
195      000224 103404 000042'
196      000232 105767 000042'
197      000240 100001
198      000242 000261
199      000244 000001
200      .END

```

5-	54	MACRO DEFINITIONS
6-	79	LOCAL DATA
7-	93	\$ALPOL - ALLOCATE FROM EXTENDED POOL
8-	157	\$DEPOL - DEALLOCATE NETWORK POOL
9-	266	\$ALIPR - ALLOCATE POOL FROM PROCESS
10-	299	\$ALL16 - ALLOCATE POOL FROM DSR
11-	328	\$DEA16 - DEALLOCATE FROM DSR
12-	365	\$ALN18 - ALLOCATE NON-UMR NETWORK POOL
12-	366	\$ALL18 - ALLOCATE FROM UMR NETWORK POOL
14-	468	\$XLINK - LINK/UNLINK BLOCK IN EXTENDED POOL

```

429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445 001306
446 001306
447 001306 032767 000000G 000000G
448 001314 001772
449
450 001316
451 001322 010003
452 001324 010105
453 001326 010046
454 001330 016704 000000G
455 001334 026703 000000G
456 001340 101402
457 001342 016704 000000G
458 001346
459 001352 026726 000000G
460 001356 101403
461 001360 010467 000000G
462 001364 000402
463 001366 010467 000000G
464 001372
465
466

: +
: $DEA22 - DE-ALLOCATE DOUBLEWORD ADDRESSABLE STORAGE ( BUFFER POOL )
:
: INPUTS:
: R0 - BIAS OF BASE OF BUFFER TO BE RETURNED
: R1 - BLOCK SIZE (MAPPED) OR BYTE SIZE (UNMAPPED) OF BUFFER
: $OBIAS - BIAS OF BASE OF NEXT FREE BUFFER
:
: OUTPUTS:
: $OBIAS - BIAS OF BASE OF NEXT FREE BUFFER
: ALL REGISTERS PRESERVED
:
: NOTE:
: IF IMPROPER VALUES ARE SUPPLIED TO THIS ROUTINE, IT WILL
: PROBABLY RESULT IN THE CORRUPTION OF THE OPERATING SYSTEM.
:
: -
: $DEA22::
: $DEA18::
: BIT #FE,CX,$FMASK ; IS COMEXEC LOADED ?
: BEQ 101$ ; NO, THEN VNP ALLOCATIONS FAIL
:
: CALL $SAVAL ; SAVE ALL REGISTERS
: MOV R0,R3 ; COPY BUFFER BIAS
: MOV R1,R5 ; COPY BLOCK COUNT
: MOV R0,-(SP) ; SAVE BIAS FOR LATER
: MOV $OBIAS,R4 ; ASSUME MAPPED AREA
: CMP $QSTRT,R3 ; IS ALLOCATION FROM MAPPED AREA ?
: BLOS 2$ ; IF LOS, YES
: MOV $NBIAS,R4 ; ELSE USE NON-MAPPED POINTER
: CALL .DMEM ; DE-ALLOCATE BACK TO NETWORK POOL
: CMP $QSTRT,(SP)+ ; WAS ALLOCATION FROM MAPPED AREA ?
: BLOS 4$ ; IF LOS, YES
: MOV R4,$NBIAS ; UPDATE NON-MAPPED POINTER
: BR 6$
: MOV R4,$OBIAS ; UPDATE MAPPED POINTER
: RETURN
:
: .DSABL 1$B

```

```

121      ;+
122      $FILE ~ ".FILE" ACTION ROUTINE
123      ;
124      INPUTS:
125      F2.FIL SET IF VALID FILE NAME SEEN ON THIS LINE
126      $FILE=ASCIZ FILE NAME
127      ;
128      OUTPUTS:
129      R0,R1,R2=DESTROYED
130      SECONDARY TEMPLATE FILE PROCESSED
131      ;
132      ;
133      .ENABL LSB
134
135      000000      $FILE::
136      000000 032767 000000G 000000G      BIT      #F2.FIL,$FLAG2 ; WAS VALID FILE NAME SEEN?
137      000006 001422      BEQ      10$      ; IF EQ, NO
138      000010 005000      CLR      R0      ; SETUP ASCIZ FILE SPEC
139      000012 012701 000000G      MOV      #$FILEN,R1 ; FOR [131,5X]FILENAME.DAT
140      000016 012702 000000'      MOV      #TMPEXT,R2 ;
141      000022      CALL      $AZFS ; ...
142      000026      CALL      $CLQUE ; TRY TO OPEN SECONDARY TEMPLATE
143      000032 105777 000000G      TSTB      @$CLIOS ; SUCCESS?
144      000036 100407      BMI      101$ ; IF MJ, NO
145      000040 016746 000000G      MOV      $VALUE,-(SP) ; SAVE EXPRESSION VALUE
146      000044      CALLR      $TMFIX ; PROCESS SECONDARY TEMPLATE
147      000050      $FILEX:: ; OVERLAY RETURN ADDRESS
148      000050 012667 000000G      MOV      (SP)+,$VALUE ; RESTORE EXPRESSION VALUE
149      000054      10$:      RETURN
150
151      ;
152      ; ERROR CONDITIONS
153      ;
154      000056      101$:      MSG$R 17 ; SECONDARY TEMPLATE NOT FOUND
155
156      .DSABL LSB

```

```

498
499          ;+ $VFMUX - VERIFY MULTIPLEXED LINE
500          ;
501          ; INPUTS:
502          ;     NONE
503          ;
504          ; OUTPUTS:
505          ;     ERROR UNLESS MULTIPLEXED LINE
506          ; -
507
508 001234          $VFMUX::
509 001234 032767 000000G 000000G  BIT    #FL.MUX,$FLAGS ; MUX LINE?
510 001242 001003          BNE    10$      ; IF NE, YES
511 001244          MSG$    81            ; NOT MULTIPLEXED LINE
512 001252
513 10$: RETURN

```

173 000206

CALLR @ (R0)+ ; PRINT ERROR MESSAGE

```

320 000654 116722 000000G      11$:  MOVB  $CVAL,(R2)+      : STORE COUNT (FOR CLEANUP ONLY)
321 000660 116712 000001G      MOVB  $CVAL+1,(R2)      :
322                                     : ...
323 000664 005767 000000G      TST    ,MGMGE          : MEMORY MANAGEMENT SYSTEM ?
324 000670 100404 000000G      BMI    12$             : NO
325                                     :
326 000672 062767 000004 000000G  ADD    #4,$SIZE        : ADJUST DATA BASE SIZE
327 000700 000403 000000G      BR     20$             :
328                                     :
329 000702 062767 000002 000000G 12$:  ADD    #2,$SIZE        : ADJUST DATA BASE SIZE
330 000710 000000G 20$:  RETURN
331                                     :
332                                     : ERROR CONDITION
333                                     :
334                                     :
335 000712      ALLERR: MSG$R 26      : RESOURCE ALLOCATION FAILURE

```

1


```

100
101
102
103
104
105
106
107
108
109
110
111
112
113 000000
114 000000 032767 000000G 000000G
115 000006 001447
116
117 000010 005767 000000G
118 000014 100406
119
120 000016 012767 000000G 000000G
121 000024 116767 000354 000000G
122
123 000032
124
125 000036 005767 000000G
126 000042 100403
127
128 000044
129 000050 000402
130
131 000052
132
133 000056 103420
134 000060 010267 000000G
135 000064 005067 000000G
136 000070 032767 000000G 000000G
137 000076 001013
138 000100
139 000104 103405
140 000106
141 000112 103005
142 000114
143 000120 042767 000000G 000000G
144 000126

+* $ABFI - ACCESS BINARY/LIBRARY FILE ACTION ROUTINE
: INPUTS:
: $FILEN=BINARY FILE NAME
: OUTPUTS:
: $BFALL=ALLOCATION SIZE IN 32. WORD BLOCKS
: $BFLBN=LOGICAL BLOCK NUMBER
: $BFPTR=BINARY BUFFER POINTER
: $BFXFR=TRANSFER SIZE IN BYTES
: F2.FIL CLEARED IF ANY ERROR OCCURS
:
$ABFI::
BIT #F2.FIL,$FLAG2 ; WAS VALID FILE NAME SEEN?
BEQ 20$ ; IF EQ, NO

TST .MGMGE ; MEMORY MANAGEMENT SYSTEM ?
BMI 4$ ; NO

MOV #.BASEB,$VALUE ; SET DEFAULT MAPPING
MOVB MAPBYT,$CAPR ; AND DEFAULT APR NUMBER

4$: CALL $VFLV1 ; MAKE SURE IT'S THE PRIMARY TEMPLATE

TST .MGMGE ; MEMORY MANAGEMENT SYSTEM ?
BMI 6$ ; NO

CALL $RQ7 ; REQUEST BINARY BUFFER
BR 8$

6$: CALL $RQ5 ; DITTO

8$: BCS 15$ ; IF CS, NO ACCESS
MOV R2,$BFPTR ; SAVE BUFFER POINTER
CLR $BFALL ; NO FILE SIZE YET
BIT #FL.CHA,$FLAGS ; MUX CHARACTERISTICS UPDATE ONLY ?
BNE 20$ ; YES
CALL BN.OPN ; OPEN BINARY FILE, GET ATTRIBUTES
BCS 15$ ; IF CS, FAILURE
CALL BN.LBL ; PROCESS LABEL BLOCK
BCC 20$ ; IF CC, SUCCESS
CALL $CLDEQ ; CLOSE BINARY FILE
BIC #F2.FIL,$FLAG2 ; CLEAR VALID FILE NAME FLAG

10$: CALL
15$:
20$: RETURN

```

```

108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124 000000
125 000000 032767 000000G 000000G
126 000006 001513
127 000010 016700 000000G
128 000014 012701 000001'
129 000020 016702 000000G
130
131
132
133
134
135
136
137
138
139
140
141
142
143 000024 112021
144
145
146 000026
147
148
149
150
151
152 000032 012702 000011
153 000036 166702 000000G
154 000042 001404
155 000044 112721 000040
156 000050 005302
157 000052 001374
158 000054 012700 000001'
159 000060
160 000064 010167 000000G
161 000070
162 000074 010167 000002G
163 000100
164 000104 010167 000004G

; *
; $LIBR1 - ".LIBR" OPERAND 1 ACTION ROUTINE
;
; THIS ROUTINE IS CALLED TO LOOK FOR AN EXISTING LIBRARY WHOSE NAME MATCHES
; THE FILE NAME FROM THE TEMPLATE LINE. IF NO MATCH IS FOUND, THE TPARS
; TRANSITION IS REJECTED.
;
; INPUTS:
; .PSTPT=TPARS STRING POINTER
; .PSTCN=TPARS COUNT
;
; OUTPUTS:
; F2.LBR SET IF A MATCH IS FOUND
; $LBIAS=LIBRARY DESCRIPTOR BIAS (MAPPED ONLY)
; $LBADR=LIBRARY DESCRIPTOR ADDRESS OR 0
;
; $LIBR1::
; BIT #F2.FIL,$FLAG2 ; WAS VALID FILE NAME SEEN?
; BEQ 65$ ; IF EQ, NO
; MOV .PSTPT,R0 ; GET TPARS POINTER
; MOV #ASCII,R1 ; LOCAL DATA ADDRESS
; MOV .PSTCN,R2 ; AND TPARS LENGTH
;
; .IF DF R$$MPL
; SAVRG <R3,R2> ; SAVE REGS
; MOV #LIBNM,R3 ; GET BUFFER ADDRESS
; MOV #10,,R2 ; GET BUFFER SIZE
; 1$: CLRB (R3)+ ; SPACE OUT BUFFER
; SOB R2,1$ ;
; RESRG <R2> ; RESTORE .PSTCN
; MOV #LIBNM,R3 ; GET BUFFER ADDRESS
;
; 5$: MOVB (R0),(R3)+ ; COPY FILE NAME
; .IFTF
;
; 10$: MOVB (R0)+,(R1)+ ; COPY FILE NAME
;
; .IFF
; SOB R2,10$ ; ...
; .IFT
; SOB R2,5$ ;
; RESRG <R3> ; RESTORE REG
; .ENDC
;
; MOV #9,R2 ; BLANK FILL OUT TO 9 CHARACTERS
; SUB .PSTCN,R2 ; ...
; BEQ 30$ ; IF EQ, NAME WAS 9 LONG
; 20$: MOVB #' ,(R1)+ ; ...
; DEC R2 ;
; BNE 20$ ;
; 30$: MOV #ASCII,R0 ; CONVERT FILE NAME TO RAD50
; CALL $CAT5 ; ...
; MOV R1,$FNR50 ; ...
; CALL $CAT5 ; ...
; MOV R1,$FNR50+2 ; ...
; CALL $CAT5 ; ...
; MOV R1,$FNR50+4 ; ...

```

VAC6 CREATED BY MACRO ON 29-JUN-85 AT 02:10 PAGE 3 H 7

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

CALL	8-159	8-161	8-163	8-170	8-172	8-176	8-179	8-181	8-183	8-185
	8-187	8-196	8-201	8-203	10-260	10-261	11-284	13-324	13-332	13-333
	13-336	14-366	14-374	14-375	14-378	15-396	16-414			
EMSG\$	#5-58	11-284	13-332	14-374	15-396					
NTLR\$	#5-58	7-97	7-98	7-99	7-100					
RESRG	#5-58									
RETURN	8-207	9-248	10-265	11-285	12-298	13-339	14-381	15-397	16-415	
SAVRG	#5-58									
SOB	8-146									

VAC7 CREATED BY MACRO ON 29-JUN-85 AT 02:10 PAGE 1 H 8

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
BASE	000402 R	6-126 #10-234
FLCHA	= ***** GX	6-109 11-267
FMT2	000317 RG	5-79 5-80 #5-87 5-89
FMT6	000276 RG	5-76 5-77 5-78 #5-86 5-89
FM.2	= 000000	#5-79 #5-80
FM.6	= 000000	#5-76 #5-77 #5-78
F1.UMR	= ***** GX	7-157
F2.FIL	= ***** GX	6-107
F2.LBR	= ***** GX	6-112 12-284
LBLK	000000 RG	#5-71 *6-111 6-120 9-214 9-220 12-287
LB.APR	= ***** GX	10-237
LB.USE	= ***** GX	*11-271
LINK	000340 R	6-122 #9-212
MAX14	000304 R	7-154 #8-190
REP6	000636 P	5-76 #13-308
SSBAS	= *****	5-76 5-77 5-78 5-79 5-80
USE	000444 R	5-80 #11-253
XFER	000120 R	6-118 #7-143
\$ALB	= ***** GX	6-116
\$ALL18	= ***** GX	7-161
\$ALN18	= ***** GX	7-159
\$BFALL	= ***** GX	7-156 7-174
\$BFBAS	= ***** GX	7-145 7-147 7-151 *10-237 10-238
\$BFPTR	= ***** GX	11-253
\$CAPR	= ***** GX	10-238 13-322
\$CLDEQ	= ***** GX	12-286
\$DEA18	= ***** GX	7-175 12-293
\$ELINE	= ***** GX	5-76 5-77 5-78 5-79 5-80
\$ERRY2	000002 R	#5-76 7-149 10-240
\$ERRY3	000076 R	#5-78 7-171
\$ERRY4	000152 R	#5-79 7-173
\$ERRY6	000220 R	#5-80 8-201
\$ERRY7	000036 R	#5-77 7-153
\$FILEN	= ***** GX	13-311
\$FLAGS	= ***** GX	6-109 11-267
\$FLAG1	= ***** GX	7-157
\$FLAG2	= ***** GX	6-107 6-112 12-284
\$GETLB	= ***** GX	9-217 10-235 11-269
\$LBADR	= ***** GX	6-114 10-236 11-258 11-259 11-270
\$LBERR	000570 RG	7-170 7-172 7-176 8-200 #12-283
\$LBIAS	= ***** GX	10-234 11-255 11-257
\$LBXFR	= ***** GX	7-163
\$LIBR	000000 RG	#6-106
\$NALL	= ***** GX	8-190
\$NLIB	= ***** GX	8-191 *11-273
\$NTLHB	= ***** GX	9-212 *9-214
\$PUTLB	= ***** GX	9-221 11-272
\$REP.C	= ***** GX	5-77 5-79 5-80
\$REP.P	= ***** GX	5-78
\$SIZE	= ***** GX	*11-264 *11-266
\$TERR	= ***** GX	5-78 5-79 5-80

```

55                                     :***
56                                     : LIBRARY MACROS
57                                     :***
58                                     .MCALL ASR$,EMSG$,NTLR$,CEACC$,PUTAD,SAVRG,RESRG,VECDF$
59                                     .ENABL LC
60
61 000000                                VECDF$                                ; VECTOR BLOCK OFFSETS
62
63                                     :***
64                                     : LOCAL DATA
65                                     :***
66
67 000000                                .PSECT DATA,D
68
69
70
71                                     .IF DF R$$MPL
72
73 LIBTAB: .ASCII /$XXXVT/                ; LIBRARY VECTOR TABLE START
74 LIBLEN: .ASCII /$XXXVL/                ; LIBRARY VECTOR TABLE LENGTH
75
76 STBEXT: .ASCIZ /STB/                    ; SYMBOL TABLE EXTENSION
77
78
79 LIBTB: .WORD LIBLN,0                    ; SYMBOL CONTROL BLOCK FOR $XXXVT
80 LIBTB5: .BLKW 2
81 TABVAL: .WORD 0
82
83 LIBLN: .WORD 0,0                        ; SYMBOL CONTROL BLOCK FOR $XXXVL
84 LIBLN5: .BLKW 2
85 LENVAL: .WORD 0
86
87 .ENDC
88
89
90                                     :
91                                     : LIBRARY ADDRESS WITHIN NTPool
92                                     :
93 000000                                BFADR: .BLKW 1
94
95                                     :
96                                     : ERROR MESSAGE (FORMAT STRING ADDRESS DEFINED GLOBALLY IN 'TMPAC7')
97                                     :
98 000002                                NTLR$ ,Y5,6,$TERR,$CUR.N,$ELINE,<Read failure ( -***. ).>
99
100 .EVEN
101 000000                                .PSECT

```

```

182 000156
183 000162 010167 000000G CALL $MUL ; SEE HOW MUCH SPACE WE NEED
MOV R1,$1CBSZ ; WILL NEED LATER FOR WRITE TO IMAGE
184
185 .IF DF R$$MPL ;[MP01]
186 ADD #4,$1CBSZ ; INCLUDE LINK POINTER AND COUNT FILED ;[MP01]
187 MOV $1CBSZ,R1 ; UPDATE ALLOCATION SIZE ;[MP01]
188
189 CALL $ALICB ; ALLOCATE ICB ;[MP01]
190 BR 25$ ; CONTINUE ;[MP01]
191 .ENDC ;[MP01]
192 000166 20$: CALL $ALL16 ; ALLOCATE SINGLEWORD STORAGE
193 000172 103450 BCS 112$ ; IF CS, FAILURE
194
195 .IF DF R$$MPL ;[MP01]
196 GETRV RO,$1CBSZ ; READ ICB ;[MP01]
197 SAVRG <R0> ; SAVE REG ;[MP01]
198 MOV #SBFR,RO ; GET BUFFER ADDRESS ;[MP01]
199 MOVB $NVECT,<R0>+ ; SET UP VECTOR COUNT ;[MP01]
200 MOVB $1CBLN,<R0>+ ; SET UP ICB SIZE ;[MP01]
201 CLR <R0> ; CLEAR LINK WORD ;[MP01]
202 RESRG <R0> ; RESTORE REG ;[MP01]
203 PUTRC ; UPDATE IMAGE ;[MP01]
204 ADD #2,RO ; POINT TO BEGINNING OF ICB ;[MP01]
205
206 .ENDC ;[MP01]
207
208
209 000174 010067 000000G 25$: MOV RO,$ICBAD ; STORE ALLOCATION ADDRESS
210 000200 010200 MOV R2,RO ; LINE TALE ADDRESS
211 000202 000405 BR 40$
212
213 000204 30$: CALL $ALL16 ; ALLOCATE SINGLEWORD STORAGE
214 000210 103436 BCS 111$ ; IF CS, FAILURE
215 000212 010067 000000G 35$: MOV RO,$ICBAD ; STORE LINE TABLE ADDRESS
216 000216 010067 000000G 40$: MOV RO,$SADDR ; SAVE LINE TABLE ADDRESS
217 000222 010067 000000G MOV RO,$TADDR ; SAVE LINE TABLE ADDRESS ( DYNAMIC )
218 000226 012767 000000G 000000G MOV #TMPBF,$ADDR ; SET INTERMEDIATE BUFFER ADDRESS
219 000234 010067 000000G MOV RO,$LINK ; COMPUTE RELOCATION BIAS FROM #TMPBF
220 000240 162767 000000G 000000G SUB #TMPBF,$LINK ; TO SCOM LINE TABLE
221 000246 CALL UBA ; CALCULATE UNLOAD BLOCK ADDRESS
222 000252 100$: RETURN
223
224
225 ; ERROR CONDITIONS
226
227 000254 012767 177777 000000G 101$: MOV #-1,$SADDR ; SET ADDRESS FOR EMPTY LLC DATA BASE
228 000262 012767 000000G 000000G MOV #TMPBF,$ADDR ; SET INTERMEDIATE BUFFER ADDRESS
229 000270 032767 000000G 000000G BIT #FL.LLC,$FLAGS ; IS THIS AN LLC?
230 000276 001365 BNE 100$ ; IF NE, YES
231 000300 MSG$R 33 ; LINE TABLE EMPTY
232 000306 MSG$R 34 ; LINE TABLE ALLOCATION FAILURE
233 000314 112$: MSG$R 88 ; CTB ALOCATION FAILURE
234
235

```

VALT CREATED BY MACRO ON 8-JUL-85 AT 22:06 PAGE 1 H 11
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
CPU	000010 R	#5-93
FL.CHA	= ***** GX	6-151
FL.DDM	= ***** GX	6-153 6-168 6-174 10-477
FL.DLC	= ***** GX	6-168
FL.LLC	= ***** GX	6-229
FMT1	000542 R	5-99 5-100 5-106 5-108 5-109 #5-116
FMT3	000545 R	5-102 5-103 5-104 5-105 5-107 #5-118
FM.1	= 000000	#5-99 #5-100 #5-106 #5-108 #5-109
FM.3	= 000000	#5-102 #5-103 #5-104 #5-105 #5-107
F.NRBD	= ***** GX	*9-452 *11-591
F.RSIZ	= ***** GX	*9-453
F.URBD	= ***** GX	*9-451 *11-549 *11-590 *11-597 *11-615 *11-625
F1.INT	= ***** GX	10-484
F1.IN1	= ***** GX	5-86 5-87 5-88 10-488
F1.IN2	= ***** GX	5-87 5-88 10-488
F1.IN3	= ***** GX	5-88 10-488
F1.LFI	= ***** GX	10-497
F1.LFL	= ***** GX	10-495
F1.MPL	= ***** GX	10-491
F1.MPT	= ***** GX	10-493
GETIND	001332 R	11-527 #12-648
ITAB	000002 R	#5-86 10-489
KRBAL	000322 R	6-159 #9-377
K.CSR	000002	*9-395
K.PRI	000000	*9-390
K.VCT	000001	*9-394
L.CTL	= ***** GX	9-400 9-415
L.DDM	= ***** GX	9-399 9-413
L.KRBA	= ***** GX	*9-426
PDVX	000000 R	#5-81 *12-660 13-669
R\$SMPL	= *****	5-60 5-68 6-185 6-195 7-248 9-386 9-402 9-418 9-431
SYSFDB	= ***** GX	*9-451 *9-452 *9-453 *11-549 *11-590 *11-597 *11-615 *11-625
S\$BAS	= *****	5-99 5-100 5-106 5-108 5-109
UBA	000662 R	6-221 #11-527
UBLTA	001400 R	11-574 11-606 #13-669
ZF.PSE	= ***** GX	6-157 6-177 10-480
Z.FLG	= ***** GX	6-157 6-177 10-480
\$ADDR	= ***** GX	*6-218 *6-228
\$ADRX	= ***** GX	11-545 11-581 11-596 11-618
\$ADRZ	= ***** GX	11-544 11-582 11-595 11-616
\$ALLPR	= ***** GX	6-170
\$ALL15	= ***** GX	9-379
\$ALL16	= ***** GX	6-192 6-213
\$ALLN18	= ***** GX	11-592
\$BFR	= ***** GX	9-384 9-451 9-452 11-590 11-591 *11-607 11-615 11-618 11-620
\$DEA16	= ***** GX	*11-622 11-625
\$ERR33	000012 R	11-633
\$ERR34	000044 R	#5-99 6-231
\$ERR35	000114 R	#5-100 6-232
		#5-102 10-504

```

345 .IF DF R$$$MPL ;[MP01]
346 .SBTTL .ALSPK - ALLOCATE SECONDARY POOL ;[MP01]
347 ;+ .ALSPK - ALLOCATE SECONDARY POOL ;[MP01]
348 ;[MP01]
349 ;[MP01]
350 ;[MP01]
351 INPUTS: ;[MP01]
352 NONE ;[MP01]
353 ;[MP01]
354 OUTPUTS: ;[MP01]
355 R0 - BLOCK BIAS OF ALLOCATED I/O PACKET ;[MP01]
356 ;[MP01]
357 ;[MP01]
358 ;[MP01]
359 ;[MP01]
360 .ALSPK::SAVRG <R5,R4,R3> ; SAVE REG'S ;[MP01]
361 MOV #<I.LGTH+77>/100,R1 ; SET SIZE TOO ALLOCATE ;[MP01]
362
363 10$: SEC ; ASSUME ALLOCATION FAILURE ;[MP01]
364 BEQ 50$ ; IF NO MORE FREE BLOCKS - BRANCH ;[MP01]
365 MOV #SPOLHD,R0 ; GET ADDRESS OF LISTHEAD ;[MP01]
366 MOV R0,R4 ; SAVE LISTHEAD ;[MP01]
367 MOV (R0), $BFR ; PREPARE TO ENTER LOOP ;[MP01]
368
369 20$: MOV R0,R2 ; SAVE ADDRESS OF CURRENT BLOCK ;[MP01]
370 MOV $BFR+2,BLSIZ ; SAVE SIZE OF PREVIOUS BLOCK ;[MP01]
371 MOV $BFR,R ; GET ADDRESS OF NEXT BLOCK ;[MP01]
372 BEQ 10$ ; IF EQ END OF FREE BLOCK CHAIN ;[MP01]
373
374 GETAD #120000,#4,$BFR ; READ FIRST TWO WORDS OF NEXT BLOCK ;[MP01]
375
376 CMP R1,$BFR+2 ; BLOCK BIG ENOUGH? ;[MP01]
377 BHI 20$ ; IF HI NO ;[MP01]
378
379 SWITCH #SECTMP ; SET BUFFER ADDRESS ;[MP01]
380 GETRV .SECFR,#2 ; READ VALUE ;[MP01]
381 SUB R1,SECTMP ; ADJUST AMOUNT OF FREE SECONDARY POOL ;[MP01]
382 SWITCH #SECTMP ; SET BUFFER ADDRESS ;[MP01]
383 PUTRC ; WRITE TO IMAGE ;[MP01]
384 SWITCH ; RESET DEFAULT ;[MP01]
385 ;[MP01]
386 ;[MP01]
387 SUB R1,$BFR+2 ; REDUCE SIZE OF FREE BLOCK ;[MP01]
388 BNE 30$ ; IF NE MORE LEFT IN BLOCK ;[MP01]
389 CMP R2,R4 ; IS THIS BEING REWRITTEN TO THE LISTHEAD ? ;[MP01]
390 BNE 25$ ; NO ;[MP01]
391 MOV $BFR,(R2) ; UPDATE APPROPRIATE LISTHEAD ;[MP01]
392 BR 40$ ;[MP01]
393
394 ;
395 ; SPECIAL CASE WHERE ALLOCATE BLOCK IS THE SAME SIZE AS THE FREE BLOCK ;[MP01]
396 ;[MP01]
397 ;[MP01]
398 25$: MOV BLSIZ,$BFR+2 ; SET SIZE OF CURRENT BLOCK ;[MP01]
399
400 PUTAD #120000,#4,R2 ; REMOVE BLOCK FROM LIST ;[MP01]
401 BR 50$ ;[MP01]

```



```

168 000070 016701 000000G 40$: MOV $RBLCK,R1 ; SAVE CURRENT BIAS INS R1 ( PREVIOUS HOLE )
169 000074 016767 000000G 000000G MOV $BFR,$RBLCK ; LINK UP
170 000102 000261 SEC ; ASSUME NO HOLE
171 000104 001431 BEQ 70$ ; END OF CHAIN !
172 000106 GETAD ; READ IN NEXT DESCRIPTOR
173 000112 016700 000002G MOV $BFR+2,R0 ; LOAD HOLE SIZE INTO R0
174 000116 160500 SUB R5,R0 ; SUBTRACT DESIRED SPACE
175 000120 103763 BLO 40$ ; TOO SMALL
176 000122 001406 BEQ 50$ ; JUST RIGHT
177 000124 016746 000000G MOV $RBLCK,-(SP) ; PUSH CURRENT BIAS ON STACK
178 000130 060016 ADD R0,(SP) ; ALLOCATE FROM TOP OF FREE SPACE
179 000132 010067 000002G MOV R0,$BFR+2 ; BASE OF ADJUSTED (SHORTENED) HOLE.
180 000136 000410 BR 60$ ; RETURN ( REWRITE DESCRIPTOR )
181
182 ; DELETE EXACT SIZE HOLE FROM LIST
183
184 000140 016746 000000G 50$: MOV $RBLCK,-(SP) ; SET BIAS OF ALLOCATED SPACE ON STACK
185 000144 016702 000000G MOV $BFR,R2 ; SET POINTER TO NEXT HOLE (IF ANY) IN
186 000150 010167 000000G MOV R1,$RBLCK ; PREVIOUS HOLE.
187 000154 010267 000000G MOV R2,$BFR ; RETURN
188
189 ; REWRITE LAST DESCRIPTOR REFERENCED AND RETURN
190
191 000160 60$: PUTAD ; REWRITE LAST DESCRIPTOR
192 000164 000241 CLC ; INDICATE SUCCESS
193 000166 012605 MOV (SP)+,R5 ; RETURN BIAS OF MEMORY ALLOCATED
194 000170 70$: RETURN
195
196

```

```

Symbol table
A$$CHK= 000000      G$$TSS= 000000      L$$11R= 000000      P$$P45= 000000      $BFR = ***** GX
A$$CPS= 000000      G$$TTK= 000000      M$$CRB= 000124      P$$WRD= 000000      $CLOPE 000000RG
A$$PRI= 000000      G$$WRD= 000000      M$$CRX= 000000      Q$$OPT= 000010      $CLOST= ***** GX
A$$TRP= 000000      IO.RLB= ***** GX  M$$FCS= 000000      R$$DER= 000000      $IOSB 000042RG 002
C$$CKP= 000000      I$$RAR= 000000      M$$MGE= 000000      R$$K11= 000001      $LBN 000034RG 002
C$$ORE= 000400      I$$RDN= 000000      M$$NET= 000000      R$$SND= 000000      $LBUF 000026RG 002
C$$RSH= 177564      K$$CNT= 177546      M$$OVR= 000000      R$$11M= 000000      $LEN 000030RG 002
D$$BUG= 177514      K$$CSR= 177546      N$$ACC= 000001      STAT 000000R 002 $PUTAD= ***** GX
D$$ISK= 000000      K$$LDC= 000000      N$$BUF= 000001      SYSFDB= ***** GX $RADDR= ***** GX
D$$L11= 000001      K$$TPS= 000074      N$$LDV= 000001      S$$WRG= 000000      $RBLCK= ***** GX
D$$YNC= 000000      LBNDPB 000012R 002 N$$MCP= 000001      S$$YSZ= 007600      $READ 000220RG
D$$YNM= 000000      LD$LP = 000000      N$$MLL= 000001      T$$KMG= 000000      $READX 000060RG
E$$XPR= 000000      L$BHRB= ***** GX N$$MOV= 000010      T$$MIN= 000000      $RLBL 000026RG
F$$LVL= 000001      L$ASG= 000000      N$$NCT= 000001      V$$CTR= 001000      $SAVVR= ***** GX
F.RSIZ= ***** GX  L$DRV= 000000      N$$PEM= 000001      X$$DBT= 000000      .MGMGE= ***** GX
G$$TPP= 000000      L$P11= 000001

```

```

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
        000246 001 (RW,I,LCL,REL,CON)
DATA 000046 002 (RW,D,LCL,REL,CON)
Errors detected: 0

```

*** Assembler statistics

```

Work file reads: 0
Work file writes: 0
Size of work file: 9457 Words ( 37 Pages)
Size of core pool: 14440 Words ( 55 Pages)
Operating system: RSX-11M/PLUS

```

```

Elapsed time: 00:00:08.13
SY:VBIO.V2,[132,134]VBIO/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VBIO

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52

.TITLE VCEAL - VNP ALLOCATION/DEALLOCATION ROUTINES
.IDENT /V05.00/

.....
COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

.....
THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

.....
THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

.....
DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

.....
MODULE DESCRIPTION:

VNP - CORE ALLOCATION/DE-ALLOCATION ROUTINES

.....
DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

.....
IDENT HISTORY:

-
1.00 27-FEB-78
VERSION 2.0 RELEASE

2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0

3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1

4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0

5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/R SX V1.0

```

468 .SBTTL $XLINK - LINK/UNLINK BLOCK IN EXTENDED POOL
469
470 **-$XLINK-LINK/UNLINK BLOCK IN EXTENDED POOL
471
472 THIS ROUTINE IS CALLED TO LINK OR UNLINK A BLOCK IN A LIST THAT MAY
473 INCLUDE ALLOCATIONS FROM EXTENDED POOL.
474
475 INPUTS:
476 TO LINK BLOCK INTO LIST:
477   MOV X,-(SP) ; ADDRESS OF BLOCK TO BE LINKED INTO LIST
478   MOV Y,-(SP) ; ADDRESS OF PREVIOUS BLOCK IN LIST
479   CALL $XLINK
480 TO UNLINK BLOCK FROM LIST:
481   MOV X,-(SP) ; ADDRESS OF PREVIOUS BLOCK IN LIST
482   MOV Y,-(SP) ; ADDRESS OF BLOCK TO BE UNLINKED FROM LIST
483   CALL $XLINK
484
485 OUTPUTS:
486 BLOCK IS LINKED/UNLINKED, ADDRESSES ARE REMOVED FROM THE STACK.
487
488 REGISTERS:
489 NO REGISTERS ARE MODIFIED
490
491
492 $XLINK::
493 001374 012767 000004 000000G MOV #4,$RSIZE ; READ IN 4 BYTES
494 001402 010046 MOV RO,-(SP) ; SAVE RO
495 001404 CEACC$ 4(SP),RO ; CONVERT TO MAPPED ADDRESS
496 001416 SWITCHI #LOCBUF ; USE LOCBUF AS INPUT BUFFER
497 001424 SWTCHO #LOCBUF ; AND OUTPUT BUFFER
498 001432 GETAD RO ; READ IN LISTHEAD FOR BLOCK
499 001442 016746 176340 MOV LOCBUF,-(SP) ; SAVE ADDRESS OF NEXT BLOCK IN LIST
500 001446 016667 000010 176332 MOV TO(SP),LOCBUF ; SET NEW LINK WORD INTO BLOCK
501 001454 PUTAD RO ; WRITE BLOCK WITH UPDATED LINK WORD
502 001464 CEACC$ LOCBUF,RO ; CONVERT TO MAPPED ADDRESS
503 001476 GETAD RO ; READ IN LINK WORD
504 001506 012667 176274 MOV (SP)+,LOCBUF ; SET NEW LINK WORD
505 001512 PUTAD RO ; WRITE UPDATED LINK WORD
506 001522 SWITCHI ; RESTORE DEFAULT INPUT
507 001530 SWTCHO ; AND OUTPUT BUFFERS
508 001536 012600 MOV (SP)+,RO ; RESTORE RO
509 001540 012616 MOV (SP)+,(SP) ; CLEAN OFF STACK (SAVING RETURN
510 001542 012616 MOV (SP)+,(SP) ; ... ADDRESS)
511
512 001544 RETURN
513
514 000001 .END

```

```

158                                     ;+
159                                     ; $INTX - ".INT" ACTION ROUTINE, PART 2
160                                     ;
161                                     ; INPUTS:
162                                     ;   $VALUE=ENTRY POINT ADDRESS
163                                     ;
164                                     ; OUTPUTS:
165                                     ;   R0,R1,R2=DESTROYED
166                                     ; -
167
168 000064                               $INTX::
169 0^0064                               CALL    $RQ3           ; REQUEST BINARY FILE
170 0C 070 103411                       BCS     10$           ; IF CS, NO ACCESS
171 000072 112722 000000G                MOVB   #.INT,(R2)+    ; STORE FUNCTION CODE
172 000076 116722 000000G                MOVB   $VALUE,(R2)+   ; STORE ENTRY POINT ADDRESS
173 000102 116712 000001G                MOVB   $VALUE+1,(R2)  ;
174 000106 062767 000016 000000G        ADD     #14,,$SIZE    ; ADJUST DATA BASE SIZE
175 000114                                     10$: RETURN
176

```

```

515      ;+
516      ; VFVCT - CHECK VECTOR AVAILABILITY
517
518      ; INPUTS:
519      ; R0=BIT TO BE SET IN $FLAG1
520      ; R1=BITS WHICH MUST NOT BE SET IN $FLAG1
521      ; R2=DISPLACEMENT FROM FIRST VECTOR
522      ; OUTPUTS:
523      ; R0,R1,R2=DESTROYED
524      ; F1.ERR SET IF VECTOR NOT AVAILABLE
525      ; $NVECT INCREMENTED
526      ; -
527
528      VFVCT::
529      CALL    $VFSPC      ; MAKE SURE THIS IS THE ONLY OCCURANCE
530      BCS     10$         ; IF CS, IT ISN'T
531      BIT     R1,$FLAG1   ; ARE ANY ILLEGAL BITS SET?
532      BNE     111$        ; IF NE, YES
533      MOV     $$SIZE,$VECT1(R2) ; STORE CURRENT DISPLACEMENT
534      ASL     R2          ; CONVERT TO A DOUBLEWORD INDEX
535      ADD     $$VECT,R2   ; GIVING REAL VECTOR ADDRESS
536      CMP     $MXVCT,R2  ; LEGAL VECTOR?
537      BLOS    121$        ; IF LOS, NO
538      BIT     #FL.CHA,$FLAGS ; MUX CHARACTER 1ICS UPDATE ONLY ?
539      BNE     5$          ; YES
540      MOV     R2,R0       ; COPY VECTOR ADDRESS
541      BIC     #77,R0      ; MAKE IT A MULTIPLE OF 100
542      ASR     5,R0        ; CONVERT TO A WORD INDEX
543                        ; BY SHIFTING RIGHT 5 PLACES
544      GETRV   R2,#4       ; READ IN VECTOR IMAGE
545      CMP     @NSITAB(R0),$BFR ; IS VECTOR IN USE?
546      BNE     131$        ; IF NE, YES
547      INC     $NVECT      ; TALLY NUMBER OF TIMES CALLED
548      5$:    INC
549      10$:   RETURN
550
551      ; ERROR CONDITIONS
552
553      111$:  MSG$R 23      ; INTERRUPT LINKAGE ERROR
554      121$:  MSG$R 24      ; VECTOR NOT IN SYSTEM
555      131$:  MSG$R 25      ; VECTOR NOT AVAILABLE
556

```

J 3

```

337
338
339
340
341
342
343
344
345
346
347
348 000720
349 000720
350 000724
351 000730 103445
352 000732 005000
353 000734 032767 000000G 000000G
354 000742 001402
355 000744 005067 000000G
356 000750 016701 000000G 5$:
357 000754 0G1417
358 000756
359 000762 103430
360 000764 005201
361 000766 042701 000001
362 000772 010167 000000G
363 000776
364 001002 103743
365 001004
366 001010 105267 000000G
367 001014 112722 000000G 10$:
368 001020 110022
369 001022 000300
370 001024 110022
371 001026 116722 000000G
372 001032 116712 000001G
373 001036 062767 000002 000000G
374 001044 20$:

+ $SCOM - ".POOL" OR ".SCOM" ACTION ROUTINE
:
: INPUTS:
: $VALUE=NUMBER OF BYTES NEEDED
:
: OUTPUTS:
: R0,R1,R2=DESTROYED
: $NALL INCREMENTED IF ALLOCATION SUCCEEDED
: F1.ERR SET IF ALLOCATION FAILED
:
$SCOM::
CALL $VFLV1 : MAKE SURE IT'S THE PRIMARY TEMPLATE
CALL $RQ5 : REQUEST BINARY FILE
BCS 20$ : IF CS, NO ACCESS
CLR R0 : ASSUME ZERO LENGTH
BIT #FL.CHA,$FLAGS : MUX CHARACTERISTICS UPDATE ONLY ?
BEQ 5$ : NO ..
CLR $VALUE : YES .. NO-OP THE ALLOCATION
MOV $VALUE,R1 : GET BYTE COUNT
BEQ 10$ : IF ZERO, DON'T ALLOCATE
CALL MAX14 : ALLOW ONLY 14. ALLOCATIONS
BCS 20$ : IF CS, NO MORE ALLOWED
INC R1 : ROUND COUNT UP TO MULTIPLE OF 2
BIC #1,R1 :
MOV R1,$VALUE : RESET COUNT VALUE
CALL $ALL16 : ALLOCATE 16 BIT ADDRESSABLE STORAGE
BCS ALLERR : IF CS, FAILURE
CALL ZER16 : ZERO 16-BIT ALLOCATION
INCB $NALL : TALLY SUCCESSFUL ALLOCATION
MOVB #.SCOM,(R2)+ : STORE FUNCTION CODE
MOVB R0,(R2)+ : STORE ADDRESS
SWAB R0 :
MOVB R0,(R2)+ :
MOVB $VALUE,(R2)+ : STORE COUNT (FOR CLEANUP ONLY)
MOVB $VALUE+1,(R2)+ :
ADD #2,$SIZE : ADJUST DATA BASE SIZE
RETURN

```



```

146                                     ;+
147                                     ; BN.OPN - OPEN BINARY FILE AND GET ATTRIBUTES
148                                     ;
149                                     ; INPUTS:
150                                     ; $FILEN=BINARY FILE NAME
151                                     ;
152                                     ; OUTPUTS:
153                                     ; C-BIT=SUCCESS/FAILURE
154                                     ; R0,R1,R2=DESTROYED
155                                     ;
156 000130 005000 BN.OPN: CLR      R0          ; SETUP ASCII FILE SPEC
157 000132 012701      MOV     # $FILEN,R1    ; FOR [131,5X]FILENAME.TSK
158 000136 005002      CLR     R2            ; ...
159 000140      CALL    $AZFS                ;
160 000144      CALL    $CLOPE                ; OPEN BINARY FILE
161 000150 105777 000000G      TSTB   @ $CLIOS    ; SUCCESS? (CLEARS C-BIT)
162 000154 100403      BMI     101$          ; IF MI, NO
163 000156 052010      BIS      (R0)+,(R0)    ; IS FILE CONTIGUOUS?
164 000160 001404      BEQ      111$          ; IF EQ, NO
165 000162      RETURN
166
167                                     ;
168                                     ; ERROR CONDITIONS
169                                     ;
170 000164 101$:  MSG$R  69          ; OPEN FAILURE
171 000172 111$:  CALL    $CLDEQ          ; CLOSE BINARY FILE
172 000176      MSG$R  71          ; FILE NOT CONTIGUOUS

```

```

LOCAL DATA
165 000110 005067 000000G CLR $LBADR ; INDICATE NO LIBRARY ADDRESS YET
166 000114 016700 000000G MOV $NTLPT,RO ; GET HOME BLOCK BIAS
167 000120 012701 000000G MOV # $NTLHB,R1 ; AND ADDRESS
168 000124 010102 MOV R1,R2 ; COPY THE ADDRESS
169 000126 062702 000000G ADD #.CXLB1,R2 ; OFFSET TO LIBRARY DESCRIPTORS
170 000132 CALL CHECK ; LOOK FOR A MATCH
171 000136 103033 BCC 50$ ; IF CC, GOT ONE!
172 000140 CALL CHECK ; TWO DESCRIPTORS IN DATA BLOCK
173 000144 103030 BCC 50$ ;
174 000146 016700 000000G MOV $NTLHB+.CXLBR,RO ; GET FIRST LIBRARY BLOCK BIAS
175 000152 001425 BEQ 50$ ; IF EQ, THERE IS NONE
176 000154 40$: CALL $GETLB ; READ IN A 32. WORD LIBRARY BLOCK
177 000160 MOV R1,R2 ; COPY BUFFER ADDRESS
178 000162 022222 CMP (R2)+,(R2)+ ; SKIP LINK WORD AND COUNT
179 000164 CALL CHECK ; LOOK FOR A MATCH
180 000170 103016 BCC 50$ ; WITH FIVE DESCRIPTORS IN THIS BLOCK
181 000172 CALL CHECK ;
182 000176 103013 BCC 50$ ;
183 000200 CALL CHECK ;
184 000204 103010 BCC 50$ ;
185 000206 CALL CHECK ;
186 000212 103005 BCC 50$ ;
187 000214 CALL CHECK ;
188 000220 103002 BCC 50$ ;
189 000222 011100 MOV (R1),RO ; GET NEXT LIBRARY BLOCK BIAS
190 000224 001353 BNE 40$ ; IF NE, KEEP LOOKING
191 000226 50$:
192 000226 032767 000000G 000000G 60$: BIT #F2.LBR,$FLAG2 ; WAS LIBRARY FOUND?
193 000234 001003 BNE 70$ ; IF NE, YES
194 000236 062716 000002 65$: ADD #2,(SP) ; REJECT TPARS TRANSITION
195 000242 000426 BR 80$ ;
196 000244 70$: CALL $VFLV1 ; MAKE SURE IT'S THE PRIMARY TEMPLATE
197 000250 005767 000000G TST $MGMGE ; MAPPED TARGET SYSTEM?
198 000254 100411 BMI 72$ ; IF MI, NO
199 000256 012767 000000G 000000G MOV #.BASEB,$VALUE ; SET DEFAULT MAPPING
200 000264 016767 000000' 000000G MOV MAPBYT,$CAPR ; AND DEFAULT APR NUMBER
201 000272 CALL $RQ7 ; REQUEST BINARY BUFFER ACCESS
202 000276 000402 BR 74$ ;
203 000300 72$: CALL $RQ5 ; DITTO
204 000304 010267 000000G 74$: MOV R2,$BFPTR ; SAVE BUFFER POINTER
205 000310 103003 BCC 80$ ; IF CC, ACCESS GRANTED
206 000312 042767 000000G 000000G BIC #F2.FIL,$FLAG2 ; ELSE CLEAR VALID FILE NAME FLAG
207 000320 80$: RETURN

```

FILEID**VAC7

```

VV      VV      AAAAAA      CCCCCCCC      77777777
VV      VV      AAAAAA      CCCCCCCC      77777777
VV      VV      AA      AA      CC      77
VV      VV      AA      AA      CC      77
VV      VV      AA      AA      CC      77
VV      VV      AA      AA      CC      77
VV      VV      AA      AA      CC      77
VV      VV      AA      AA      CC      77
VV      VV      AAAAAAAA      CC      77
VV      VV      AAAAAAAA      CC      77
VV      VV      AA      AA      CC      77
VV      VV      AA      AA      CC      77
VV      VV      AA      AA      CCCCCCCC      77
VV      VV      AA      AA      CCCCCCCC      77

```

....
....
....

```

LL      SSSSSSSS      TTTTTTTTTT
LL      SSSSSSSS      TTTTTTTTTT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SSSSSS      TT
LL      SSSSSS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SS      TT
LL      SSSSSSSS      TT
LLLLLLLLLL      SSSSSSSS      TT
LLLLLLLLLL      SSSSSSSS      TT

```

VAC7 CREATED BY MACRO ON 29-JUN-85 AT 02:10 PAGE 2 I 8
SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
\$TWAR	= ***** GX	5-76 5-77
\$VALUE	= ***** GX	7-147 11-262 11-263
.CXLBR	= ***** GX	9-212 *9-214
.LIBR	= ***** GX	11-254
.MGMGE	= ***** GX	6-124 7-143 11-260 12-290

```

103
104
105
106
107
108
109
110
111
112
113
114 000000
115 000000 010067 000000'
116 000004 016767 000000G 000000G
117 000012 016767 000002G 000002G
118 000020 016767 000000G 000000G
119 000026
120 000032 103445
121 000034
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159

;+ $LBXFR - TRANSFER LIBRARY FILE INTO CORE
;
; INPUTS:
;   RO      = LIBRARY BIAS
;   $BFLBN  = LOGICAL BLOCK NUMBER
;   $BFXFR  = TRANSFER SIZE IN BYTES
;
; OUTPUTS:
;   C-BIT=SUCCESS/FAILURE
;
$LBXFR::
    MOV     RO,BFADR      ; SAVE LIBRARY ADDRESS
    MOV     $BFLBN,$LBN   ; COPY LOGICAL BLOCK NUMBER
    MOV     $BFLBN+2,$LBN+2
    MOV     $BFXFR,$LLEN  ; AND TRANSFER SIZE IN BYTES
    CALL    $READX        ; READ INTO EXTENDED MEMORY
    BCS     101$          ; IF CS, READ FAILURE
    CALL    $CLDEQ        ; CLOSE LIBRARY FILE

    .IF DF R$$$MPL

    MOV     $LIBNM,LIBTAB+1 ; COPY FIRST THREE CHARACTERS OF LIBRARY NAME
    MOV     $LIBNM,LIBLEN+1 ; ...
    MOV     $LIBNM+1,LIBTAB+2 ; ...
    MOV     $LIBNM+1,LIBLEN+2 ; ...
    MOV     $LIBNM+2,LIBTAB+3 ; ...
    MOV     $LIBNM+2,LIBLEN+3 ; ...

    MOV     #LIBTAB,RO     ; SET UP SYMBOL NAME
    CALL    $CAT5          ; CONVERT TO RAD50
    MOV     R1,LIBTB5      ; STORE VALUE
    CALL    $CAT5          ; CONVERT TO RAD50
    MOV     R1,LIBTB5+2    ; STORE RAD50 VALUE OF LAST 3 CHARACTERS
    MOV     #LIBLEN,RO     ; SET UP SYMBOL NAME
    CALL    $CAT5          ; CONVERT TO RAD50
    MOV     R1,LIBLN5      ; STORE RAD50 VALUE
    CALL    $CAT5          ; CONVERT TO RAD50
    MOV     R1,LIBLN5+2    ; STORE RAD50 VALUE OF LAST 3 CHARACTERS

    CLR     RO             ; GET DEFAULT UIC
    MOV     $LIBNM,R1      ; GET LIBRARY FILENAME
    MOV     $STEXT,R2      ; GET FILE EXTENSION
    CALL    $AZFS          ; SET UP FILE SPEC

    MOV     #LIBTB,RO      ; GET START OF SYMBOL BLOCKS
    CALL    $SYMBL         ; LOOKUP SYMBOLS
    BCS     101$

    MOV     LENVAL,$LBVL   ; COPY $XXXVL VALUE
    BEQ     5$             ; IF NOT VECTORED - BRANCH
    MOV     TABVAL,$LBVT   ; COPY $XXXVT VALUE
    BIC     #160000,$LBVT  ; CLEAR APR BITS
    BIS     #140000,$LBVT  ; MAP THRU APR6
    MOV     BFADR,$LBBLK   ; COPY LIBRARY BIAS
    CALL    ALVLIB         ; SET UP LIBRARY VECTOR TABLE

```

```

237 .SBTTL $ALICB - ALLOCATE ICB
238
239 + $ALICB - ALLOCATE ICB FOR M+ ;[MP01]
240
241
242 INPUTS:
243 $ICBSZ - SIZE OF ICB
244
245 OUTPUTS:
246 R0 - ADDRESS OF ICB
247 -
248 ;[MP01]
249 $ALICB::IF DF R$$MPL
250 SAVRG <R2> ; SAVE REG
251 SAVRG <R3> ; SAVE REG
252 TST MPRO ; MULTIPROCESSOR SYSTEM ?
253 BPL 20$ ; IF YES - BRANCH
254
255 MOV # $ICAVL,R0 ; GET ADDRESS-2
256 BIT #DF.DAS,$DFLAG ; D-SPACE SYSTEM ?
257 BNE 5$ ; IF YES - BRANCH
258 MOV # $CRAVL,R0 ; IF NOT - ALLOCATE FROM $CRAVL LIST HEAD
259
260 5$: MOV $ICBSZ,R1 ; GET ICB SIZE
261 CALL ALOC1 ; ALLOCATE ICB
262 BCC 10$ ; IF SUCCESS - BRANCH
263 EMMSG$R 9D ; IF FAILURE - PRINT ERROR
264
265 10$: GETRV R0,$ICBSZ ; READ ICB
266 MOV R0,-(SP) ; SAVE ICB ADDRESS
267 MOV # $BFR,R0 ; STORE START OF BUFFER
268 MOV $NVECT,(R0)+ ; STORE THE NUMBER OF VECTORS
269 MOV $ICBLN,(R0)+ ; STORE ICB LENGTH
270 CLR (R0) ; CLEAR LINK WORD
271 PUTRC ; WRITE OUT ICB TO IMAGE
272
273 MOV $SLTA,R1 ; STORE SLT ADDRESS
274 MOV L.KRBA(R1),R1 ; GET KRB ADDRESS
275 MOV (SP),R2 ; COPY ICB ADDRESS
276 ASR R2 ; DIVIDE ADDRESS BY 2
277 ADD #K.PRM,R1 ; INCLUDE NEGATIVE OFFSETS
278
279 GETRV R1,$KRBLEN ; READ KRB
280 MOV # $BFR,R0 ; POINT AT BUFFER
281 SUB #K.PRM,R0 ;
282 MOV R2,K.PRM(R0) ; STORE ICB ADDRESS/2 IN THE KRB
283 INC K.PRM(R0) ; SKIP ICB SIZE/COUNT FIELDS
284 PUTRC ; WRITE KRB TO IMAGE
285
286 RESRG <R0> ; RESTORE REG (RETURN ICB ADDRESS)
287 ADD #2,R0 ; POINT AT ICB START
288 RESRG <R3> ; RESTORE REG
289 RESRG <R2> ; RESTORE REG
290 RETURN

```



```
402          30$: PUTAD #120000,#4,R0 ; WRITE UPDATED BLOCK DESCRIPTOR ;[CMP01]
403
404          40$: ADD $BFR+2,R0 ; CALCULATE ADDRESS OF ALLOCATED BLOCK ;[MP01]
405          50$: RESRG <R3,R4,R5> ; RESTORE REG'S ;[MP01]
406
407          RETURN ;[CMP01]
408
409          .ENDC
410
```



```

198                                     .SBTTL .DMEM - DEALLOCATE MEMORY FROM VIRTUAL NTPool
199
200
201                                     +
202                                     .DMEM      - SUBROUTINE TO FREE A CONTIGUOUS SEGMENT OF VIRTUAL
203                                     MEMORY FROM A PARTITION.
204
205                                     INPUTS:
206                                     R3 - BASE ADDRESS OF SEGMENT TO BE RETURNED
207                                     R4 - CURRENT VALUE OF $QBIAS
208                                     R5 - SIZE OF SEGMENT IN 32 WORD BLOCKS
209
210                                     OUTPUTS.
211
212                                     MEMORY IS RETURNED TO NTPool
213                                     R4 - NEW VALUE OF $QBIAS
214                                     R0,R1,R2 -DESTROYED
215
216                                     CAUTION:
217
218                                     THIS ROUTINE CAN BE SUPPLIED WITH ANY ARGUMENTS AND IT WILL
219                                     SIMPLY ADD WHATEVER SEGMENT OF CORE THAT DEFINES TO THE FREE
220                                     SPACE LIST.
221                                     $QBIAS MUST BE UPDATED IN IT'S GLOBAL STORAGE LOCATION.
222
223
224 000172 010367 000000G .DMEM:: MOV R3,$RBLCK ; MAKE FREE SPACE TO BE RETURNED
225 000176 010567 000002G MOV R5,$BFR+2 ; LIKE A LINK IN THE FREE SPACE
226 000202 010467 000000G MOV R4,$BFR ; CHAIN.
227 000206 PUTAD #0,#4 ; WRITE GUESS TO FILE
228 000226 005704 TST R4 ; TEST $QBIAS
229 000230 001402 BEQ 20$ ; NO FREE SPACE, ADD THIS AS FIRST HOLE
230 000232 020304 CMP R3,R4 ; SHOULD THIS BE A NEW FIRST HOLE ?
231 000234 1C1001 BHI 30$ ; NO, GO FIND SLOT
232
233 ; NEW FIRST HOLE BEING CREATED
234
235 000236 010304 20$: MOV R3,R4 ; RETURN BIAS AS NEW $QBIAS VALUE
236
237 000240 010467 000000G 30$: MOV R4,$RBLCK ; MAP TO FIRST HOLE AND
238 000244 GETAD ; READ IN NEXT DESCRIPTOR
239
240 ; SCAN FOR NEXT AVAILABLE SLOT
241
242 000250 016700 000000G 40$: MOV $BFR,R0 ; STORE NEXT HOLE POINTER IN R0
243 000254 001410 BEQ 50$ ; END OF FREE SPACE !!
244 000256 020367 000000G CMP R3,$BFR ; ARE WE IN PROPER SLOT YET ?
245 000262 103405 BLO 50$ ; YES, GO CHECK BOUNDARY CONDITIONS
246 000264 010067 000000G MOV R0,$RBLCK ; LINK UP
247 000270 GETAD ; READ IN NEXT DESCRIPTOR
248 000274 000765 BR 40$ ; AND CONTINUE SCAN.
249
250 ; CHECK BOUNDARY CONDITIONS
251
252 000276 016746 000000G 50$: MOV $RBLCK,-(SP) ; CHECK LOW
253 000302 066716 000002G ADD $BFR+2,(SP) ; SIDE CONTIGUITY.
254 000306 022603 CMP (SP)+,R3 ; CONTIGUOUS ?

```

VBIO CREATED BY MACRO ON 29-JUN-85 AT 02:12 PAGE 1 I 14

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
F.RSIZ	= ***** GX	*8-167
IO.RLB	= ***** GX	5-78
IS\$AS	= *****	8-154
LBNDPB	= 000012 R	#5-77 9-187
L\$BHRB	= ***** GX	7-130
R\$SEIS	= *****	8-154
R\$S11D	= *****	8-154
STAT	= 000000 R	#5-72 6-108 6-110
SYSFDB	= ***** GX	*8-167
\$BFR	= ***** GX	8-159
\$CLOPE	= 000000 RG	#6-107
\$CLOST	= ***** GX	6-109
\$IOSB	= 000042 RG	5-81 #5-92 9-189
\$LBN	= 000034 RG	*5-86 *6-111 *6-112 *7-130 *7-131 *8-165 *8-166
\$LEUF	= 000026 RG	*5-83 *9-186
\$LEN	= 000030 RG	*5-84 *7-127 8-156 *8-157 *8-162 8-167
\$PUTAD	= ***** GX	8-167
\$RADDR	= ***** GX	*8-167
\$RBLCK	= ***** GX	*8-167
\$READ	= 000220 RG	7-128 8-163 #9-185
\$READX	= 000060 RG	#8-147
\$RLBL	= 000026 RG	#7-126
\$SAVVR	= ***** GX	8-148
.MGMGE	= ***** GX	8-150

```

54                                     .SBTTL  MACRO DEFINITIONS
55                                     ;
56                                     ; LIBRARY MACRO CALLS
57                                     ;
58                                     .MCALL  SWITCH,GETRV,ASL$,ASR$,PUTRC,PCBDF$,GETRC,PUTAD,GETAD,SWTCHO
59                                     .MCALL  CEACCS$
60
61                                     PCBDF$                                ; DEFINE PCB OFFSETS
62 000000
63
64                                     ;
65                                     ; LOCAL MACRO DEFINITIONS
66                                     ;
67                                     .MACRO  SAVMAP
68                                     MOV     $RBLCK,-(SP)
69                                     .ENDM   SAVMAP
70
71                                     .MACRO  RESMAP
72                                     MOV     (SP)+,$RBLCK
73                                     .ENDM   RESMAP
74
75                                     .MACRO  MAP_BIAS
76                                     MOV     BIAS,$RBLCK
77                                     .ENDM   MAP

```

AS.DEL= 000010	F.RSIZ= ***** GX	N\$MCP= 000001	P.REL 000014	\$CEACC= ***** GX
AS.EXI= 000004	F.URBD= ***** GX	N\$MML= 000001	P.SIZE 000016	\$CEAVL= ***** GX
AS.RED= 000001	G\$STPP= 000000	N\$MOV= 000010	P.STAT 000030	\$CRAVL= ***** GX
AS.WRT= 000002	G\$TSS= 000000	N\$NCT= 000001	P.SUB 000010	\$DEA16 001126RG
A\$CHK= 000000	G\$TTK= 000000	N\$PEM= 000001	P.SWSZ 000022	\$DEA18 001306RG
A\$CPS= 000000	G\$WRD= 000000	PS.APR= 000007	P.TCB 000026	\$DEA22 001306RG
A\$PRF= 000000	I\$RAR= 000000	PS.CHK= 010000	P.WAIT 000020	\$DEPOL 000274RG
A\$TRP= 000000	I\$RDN= 000000	PS.CKP= 040000	Q\$OPT= 000010	\$EXSIZ= ***** GX
A.IOC 000003	K\$CNT= 177546	PS.CKR= 020000	R\$DER= 000000	\$FMASK= ***** GX
A.LGTH= 000014	K\$CSR= 177546	PS.COM= 000200	R\$K11= 000001	\$GETAD= ***** GX
A.MPCT 000011	K\$LDC= 000000	PS.DEL= 000010	R\$SND= 000000	\$NBIA= ***** GX
A.PCB 000012	K\$TPS= 000074	PS.DRV= 000020	R\$T1M= 000000	\$PDVA = ***** GX
A.PCBL 000000	L\$P= 000000	PS.FXD= 004000	SYSFDB= ***** GX	\$PRAVL= ***** GX
A.PRI 000002	LDCBF1 000012R	PS.LIO= 001000	\$SWRG= 000000	\$PUTAD= ***** GX
A.STAT 000010	LOCBF2 000016R	PS.NSF= 000400	\$SYZ= 007600	\$QBIAS= ***** GX
A.TCB 000004	LOCBUF 000006R	PS.OUT= 100000	\$TKMG= 000000	\$QSTR= ***** GX
A.TCBL 000006	L\$ASG= 000000	PS.PER= 002000	T\$MIN= 000000	\$RADDR= ***** GX
C\$CKP= 000000	L\$DRV= 000000	PS.PIC= 000100	V\$CTR= 001000	\$RBLCK= ***** GX
C\$DRE= 000400	L\$P11= 000001	PS.SYS= 000040	XAVL 000002R	\$RSIZE= ***** GX
C\$RSH= 177564	L\$11R= 000000	P\$P45= 000000	X\$DBT= 000000	\$SAVAL= ***** GX
DEAC 001162R	M\$CRB= 000124	P\$WRD= 000000	Z.AVL = ***** GX	\$SAVRG= ***** GX
D\$BUG= 177514	M\$CRX= 000000	P.BLKS 000016	\$ALLPR 001024RG	\$XAVL = ***** GX
D\$ISK= 000000	M\$FCS= 000000	P.BUSY 000024	\$ALL15 001120RG	\$XLINK 001374RG
D\$L11= 000001	M\$MGE= 000000	P.IOC 000003	\$ALL16 001076RG	.ALLPR= ***** GX
D\$YNC= 000000	M\$NET= 000000	P.LNK 000000	\$ALL18 001232RG	.ALOCB= ***** GX
D\$YNM= 000000	M\$DVR= 000000	P.MAIN 000012	\$ALL22 001232RG	.ALOC1= ***** GX
E\$XPR= 000000	N\$ACC= 000001	P.NAM 000004	\$ALN18 001170RG	.AMEM = ***** GX
FE.CEX= ***** GX	N\$BUF= 000001	P.DWN 000026	\$ALPOL 000022RG	.DEACT= ***** GX
F\$LVL= 000001	N\$LDV= 000001	P.PRI 000002	\$BFR = ***** GX	.DMEM = ***** GX
F.NRBD= ***** GX				

. ABS. 000032 000 (RW,I,GBL,ABS,OVR)
001546 001 (RW,I,LCL,REL,CDN)
Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 10255 Words (41 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:19.43
SY:VCEAL.V2,[132,134]VCEAL/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VCEAL

```

178
179
180
181
182
183
184
185
186
187
188
189 000116
190 000116
191 000122 012700 000000G
192 000126 012701 000000C
193 000132 005002
194 000134
195

;+
; $INT - ".INT" ACTION ROUTINE
;
; INPUTS:
;     NONE
;
; OUTPUTS:
;     R0,R1,R2=DESTROYED
;     $VECT1=CURRENT VALUE OF $SIZE
;-

$INT::
    CALL    $VFDDM          ; MAKE SURE IT'S A DDM
    MOV     #F1.INT,R0      ; BIT TO BE SET
    MOV     #F1.IN1!F1.IN2!F1.IN3,R1 ; BITS WHICH MUST NOT BE SET
    CLR     R2              ; VECTOR INDEX
    CALLR   VVVCT           ; CHECK VECTOR AVAILABILITY

```

```

558                                     ;+
559                                     ; REP2B - REPLACE PROCESS TYPE AND VECTOR VALUE
560                                     ;
561                                     ; INPUTS:
562                                     ;   R2=VECTOR VALUE
563                                     ;   R3=FORMAT STRING ADDRESS
564                                     ;   R4=ERROR MESSAGE BUFFER ADDRESS
565                                     ;
566                                     ; OUTPUTS:
567                                     ;   R0,R1,R2=DESTROYED
568                                     ;   R3,R4=UPDATED
569                                     ; -
570
571 001420 REP2B:: CALL $REP.C ; REPLACE PROCESS NAME
572 001420 10$: MOV B (R3)+,(R4) ; LOOK FOR SECOND ASTERISK
573 001424 CMP B #'*,(R4)+ ;
574 001426 122724 000052 BNE 10$ ; ...
575 001432 001374 DEC R4 ; SKIP THE ASTERISK
576 001434 005304 MOV R4,R0 ; COPY BUFFER ADDRESS
577 001436 010400 MOV R2,R1 ; COPY VECTOR VALUE
578 001440 010201 CLR R2 ; SUPPRESS LEADING ZEROES
579 001442 005002 CALL $CBOMG ; CONVERT TO OCTAL
580 001444 MOV R0,R4 ; SET NEW BUFFER ADDRESS
581 001450 010004 RETURN
582 001452
583
584 000001 .END
585

```

```

ASYM      000000R      002 F$$LVL= 000001      L$$DRV= 000000      P$$WRD= 000000      $ELINE= ***** GX
A$$CHK= 000000      F1$TE= ***** GX      L$$P11= 000001      Q$$OPT= 000010      $ERR27 000032R      002
A$$CPS= 000000      GBLK 000006R      002 L$$11R= 000000      REP2A 000210R      $ERR29 000054R      002
A$$PRI= 000000      GETAB 000022R      002 M$$CRB= 000124      R$$DER= 000000      $ERR30 000114R      002
A$$TRP= 000000      GSYM 000012R      002 M$$CRX= 000000      R$$K11= 000001      $ERR31 000154R      002
C$$CKP= 000000      GVAL 000016R      002 M$$FCS= 000000      R$$SND= 000000      $ERR32 000204R      002
C$$ORE= 000400      G$$TPP= 000000      M$$MGE= 000000      R$$11M= 000000      $FLAG1= ***** GX
C$$RSH= 177564      G$$TSS= 000000      M$$NET= 000000      STBEXT 000312R      $GLOBL 000000RG
D$$BUG= 177514      G$$ITK= 000000      M$$OVR= 000000      S$$WRG= 000000      $PRV.N= ***** GX
D$$ISK= 000000      G$$WRD= 000000      N$$ACC= 000001      S$$YSZ= 007600      $REP.C= ***** GX
D$$L11= 000001      I$$RAR= 000000      N$$BUF= 000001      T$$KMG= 000000      $REP.P= ***** GX
D$$YNC= 000000      I$$RDN= 000000      N$$LDV= 000001      T$$MIN= 000000      $SYMBL= ***** GX
D$$YNM= 000000      K$$CNT= 177546      N$$MCP= 000001      V$$CTR= 001000      $TERR = ***** GX
E$$XPR= 000000      K$$CSR= 177546      N$$MML= 000001      X$$DBT= 000000      $TWARN= ***** GX
FMT2A     000232R      002 K$$LDC= 000000      N$$MOV= 000010      $AZFS = ***** GX
FMT4      000262R      002 K$$TPS= 000074      N$$NCT= 000001      $CATS = ***** GX
FM.2A     = 000000      LD$LP = 000000      N$$PEM= 000001      $CLEVL= ***** GX
FM.4      = 000000      L$$ASG= 000000      P$$P45= 000000

. ABS.     000000      000 (RW,I,GBL,ABS,OVR)
          000270      001 (RW,I,LCL,REL,CON)
DATA      000316      002 (RW,D,LCL,REL,CON)
Errors detected: 0

```

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 10034 Words (40 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:09.34

SY:VAC3.V2,[132,134]VAC3/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VAC3

```

376
377
378
379
380
381
382
383
384
385
386
387 001046 116700 000000G
388 001052 010046
389 001054 116700 000000G
390 001060 062600
391 001062 022700 000015
392 001066 103401
393 001070
394
395
396
397
398 001072

;+
; MAX14 - CHECK FOR NO MORE THAN 14. ALLOCATIONS
;
; INPUTS:
; $NALL=NUMBER OF ALLOCATIONS
; $NLIB=NUMBER OF LIBRARIES
;
; OUTPUTS:
; C-BIT=SUCCESS/FAILURE
; RO=DESTROYED
;
MAX14:  MOVB  $NALL,RO      ; GET NUMBER OF ALLOCATIONS
        MOV   RO,-(SP)    ; SAVE IT
        MOVB  $NLIB,RO    ; GET NUMBER OF LIBRARIES
        ADD   (SP)+,RO    ; ADD THE TWO TOGETHER
        CMP   #13,RO      ; ARE WE AT THE MAXIMUM ALREADY?
        BCS   101$        ; IF CS, NO MORE ALLOWED
        RETURN
;
; ERROR CONDITION
;
101$:   MSG$R  78          ; ONLY 14. ALLOCATIONS/LIBRARIES ALLOWED

```



```

174
175
176
177
178
179
180
181
182
183
184
185
186
187
188 000204 012700 000000' BN.LBL: MOV #DSKBUF,R0 ; GET DISK BUFFER ADDRESS
189 000210 CALL $RLBL ; READ LABEL BLOCK
190 000214 103457 BCS 101$ ; IF CS, FAILURE
191 000216 016767 000000G 000000G MOV $LBN,$BFLBN ; COPY LOGICAL BLOCK NUMBER
192 000224 016767 000002G 000002G MOV $LBN+2,$BFLBN+2 ;
193 000232 016067 000000G 000000G MOV $BSA(R0),$FBAS ; SAVE BASE ADDRESS
194 000240 032760 000000G 000000G BIT #TS$NHD,$BFLG(R0) ; DOES FILE HAVE A HEADER?
195 000246 001450 BEQ 111$ ; IF EQ, YES - ERROR
196 000250 016001 000000G MOV $BLDZ(R0),R1 ; GET FILE SIZE IN BLOCKS
197 000254 026001 000000G CMP $B: '7(R0),R1 ; IS FILE OVERLAID?
198 000260 001043 BNE 111$ ; IF NE, YES - ERROR
199 000262 022701 001777 CMP #1777,R1 ; IS FILE SIZE > 32K-32 WORDS?
200 000266 103443 BLO 121$ ; IF LO, YES
201 000270 010102 MOV R1,R2 ; COPY FILE SIZE
202 000272 066002 000000G ADD $BEXT(R0),R2 ; ADD EXTENSION SIZE
203 000276 022702 002000 CMP #2000,R2 ; IS EXTENSION > 32K WORDS?
204 000302 103440 BLO 131$ ; IF LO, YES
205 000304 ASL$ 6,R1 ; CONVERT SIZE FROM BLOCKS TO BYTES
206 000320 010167 000000G MOV R1,$BFXFR ; SET TRANSFER SIZE IN BYTES
207
208 000324 005767 000000G TST .MGMGE ; MEMORY MANAGEMENT SYSTEM ?
209 000330 100006 BPL 5$ ; NO
210
211 000332 ASL$ 6,R2 ; CONVERT EXTENSION FROM BLOCKS TO BYTES
212
213 000346 010267 000000G 5$: MOV R2,$BFALL ; SAVE ALLOCATION SIZE NEEDED
214 000352 10$: RETURN
215
216
217
218
219 000354 116777 000000G 000000G 101$: MOV $IOSB,$CLIOS ; COPY ERROR CODE
220 000362 MSG$R 72 ; LABEL BLOCK READ FAILURE
221 000370 111$: MSG$R 73 ; INVALID BINARY IMAGE
222 000376 121$: MSG$R 74 ; SIZE EXCEEDS 32K-32 WORDS
223 000404 131$: MSG$R 75 ; EXTENSION EXCEEDS 32K WORDS
224
225 000001 .END

```

```

LOCAL DATA

209
210      ;+ CHECK - CHECK THIS LIBRARY DESCRIPTOR FOR A MATCH
211
212      INPUTS:
213          R0=LIBRARY BLOCK BIAS
214          R1=ADDRESS OF 32. WORD BLOCK
215          R2=CURRENT POINTER INTO 32. WORD BLOCK
216          $FNR50=RAD50 FILE (LIBRARY) NAME TO BE MATCHED
217
218      OUTPUTS:
219          C-BIT=SUCCESS/FAILURE
220          R2 INCREMENTED BY 12. (SIZE OF LIBRARY DESCRIPTOR)
221
222      CHECK: MOV     R0,$LBIAS      ; SAVE BIAS OF THIS BLOCK
223              TST     (R2)         ; IS THIS AN EMPTY SLOT?
224              BEQ     10$          ; IF EQ, YES - SAVE ITS ADDRESS
225              CMP     (R2),$FNR50  ; COMPARE THREE WORDS
226              BNE     20$          ; ...
227              CMP     2(R2),$FNR50+2 ; ...
228              BNE     20$          ; ...
229              CMP     4(R2),$FNR50+4 ; ...
230              BNE     20$          ; ...
231              TST     ,MGMGE       ; MAPPED TARGET SYSTEM?
232              BMI     5$           ; IF MI, NO
233              BIT     #FL.CHA,$FLAGS ; MUX CHARACTERISTICS UPDATE ONLY ?
234              BNE     5$           ; YES
235              BIT     #F1.UMR,$FLAG1 ; IS UMR MAPPING NEEDED ?
236              BEQ     4$           ; IF EQ, NO
237              CMP     LB,ADR(R2),$QSTRT ; IS THIS COPY IN MAPPED SPACE ?
238              BLO     20$          ; IF LO, NO - CAN'T USE THIS COPY
239              CMPB    LB,USE(R2),#377 ; IS USE COUNT AT MAX VALUE?
240              BEQ     20$          ; IF EQ, YES - CAN'T USE THIS ONE
241              BIS     #F2.LBR,$FLAG2 ; WE FOUND IT!
242              MOV     R2,$LBADR     ; SAVE OFFSET INTO BLOCK OF THIS DESCRIPTOR
243              SUB     R1,$LBADR     ; ...
244              ADD     #12,R2        ; ADVANCE R2 TO NEXT DESCRIPTOR (CLEARS C-BIT)
245              BIT     #F2.LBR,$FLAG2 ; DID WE FIND A MATCH?
246              BNE     30$          ; IF NE, YES
247              SEC     30$          ; INDICATE FAILURE
248              RETURN

```

.TITLE VAC7 - VNP TEMPLATE ACTION ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - TEMPLATE SYNTAX TPARS ACTION ROUTINES, PART 7

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

VAC7 CREATED BY MACRO ON 29-JUN-85 AT 02:10 PAGE 3 J 8

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

CALL	6-116	6-118	6-122	6-126	6-127	6-129	7-149	7-153	7-154	7-159
	7-161	7-163	7-170	7-172	7-175	7-176	8-200	9-217	9-221	10-235
	10-240	11-269	11-272	12-286	12-293					
EMSG\$	#5-59	7-149	7-153	10-240						
EMSG\$R	#5-59	7-171	7-173	8-201						
NTLRS\$	#5-59	5-76	5-77	5-78	5-79	5-80				
RETURN	6-130	7-165	7-178	8-195	9-222	10-241	11-274	12-294	13-323	

```

160                                     5$:
161                                     .ENDC
162
163 000040 016700 000000G      MOV    $LBIAS,R0      ; GET DESCRIPTOR BIAS
164 000044                  CALL  $GETLB           ; READ LIBRARY BLOCK
165 000050 026700 000000G      CMP    $NTLPT,R0     ; IS IT THE HOME BLOCK?
166 000054 001402                  BEQ    10$        ; IF EQ, YES
167 000056 005261 000002      INC    2(R1)         ; TALLY DESCRIPTORS/BLOCK
168 000062 066701 000000G      10$: ADD    $LBADR,R1      ; OFFSET TO DESCRIPTOR
169 000066 016721 000000G      MOV    $FNR50,(R1)+   ; COPY RAD50 NAME
170 000072 016721 000002G      MOV    $FNR50+2,(R1)+ ;
171 000076 016721 000004G      MOV    $FNR50+4,(R1)+ ;
172 000102 016721 000000G      MOV    BFADR,(R1)+    ; LIBRARY ADDRESS
173 000106 016721 000000G      MOV    $BFALL,(R1)+   ; AND LIBRARY SIZE
174 000112 005021                  CLR    (R1)+       ; ZERO THE USE COUNT
175 000114 005767 000000G      TST    ,MGMGE        ; MAPPED TARGET SYSTEM?
176 000120 100410                  BMI    20$        ; IF MI, NO
177 000122 016702 000000G      MOV    $BFBAS,R2      ; GET BASE ADDRESS
178 000126 006002                  ROR    R2          ; CONVERT TO AN APR NUMBER
179 000130                  ASR$    4,R2             ;
180 000140 010241      MOV    R2,-(R1)              ; STORE ii
181 000142 000241      20$: CLC                     ; RETURN C-BIT CLEAR
182 000144                  RETURN
183
184                                     ;
185                                     ; ERROR CONDITION
186
187 000146 116777 000000G 000000G 101$: MOVB    $IOSB,$CLIOS ; COPY I/O ERROR CODE
188 000154                  EMMS$    Y5              ; LIBRARY FILE READ ERROR
189 000162 016700 000000G      MOV    BFADR,R0       ; RETURN WITH LIBRARY ADDRESS IN R0
190 000166                  RETURN                    ; AND C-BIT SET
191

```

```

292
293
294
295          20$:  MOV    $SLTA,R1      ; GET SLT ADDRESS
296              MOV    L,KRBA(R1),R1 ; GET KRB ADDRESS
297              ADD    #K.PRM,R1      ; ADJUST R1 TO OFFSET 0
298
299              GETRV   R1,#$KRBLN     ; READ KRB
300              MOV    #BFR,R0        ; SET UP BUFFER ADDRESS
301              SUB    #K.PRM,R0      ; POINT TO OFFSET 0
302              MOV    K.URM(R0),R0   ; GET UNIBUS RUN MASK
303              MOV    #-1,R3         ; INITIALIZE CPU NUMBER * 2
304
305          30$:  ADD    #1,R3          ; ... THIS OUR CPU ?
306              ASR    R0              ; IF NO - BRANCH
307              BCC    30$             ; CLEAR C-BIT
308              CLC                     ; COPY CPU NUMBER
309              MOV    R3,R0
310
311              SWAB   R0              ; POSITION CPU NUMBER
312              ASL    R0              ; ...
313              ASL    R0              ; ...
314              ASL    R0              ; ...
315              ASL    R0              ; ...
316              MOV    R0,CPU         ; STORE VALUE
317
318              MOV    #ICAVL,R0       ; GET ICB AVAILABLE LIST FOR FIRST PROCESSOR
319
320              SAVRG  <R1>            ; SAVE KRB ADDRESS
321              MOV    R3,R1           ; GET CPU NUMBER
322              ASL    R1              ; MULTIPLY BY 8.
323              ASL    R1              ; ...
324              ASL    R1              ; ...
325              ADD    R1,R0           ; CALCULATE CORRECT ICB LISTHEAD
326
327              MOV    $ICBSZ,R1       ; GET ICB LENGTH
328              MOV    R3,R2           ; COPY CPU NUMBER
329              ASL    R2              ; CONVERT TO WORD OFFSET
330              ADD    .K6TAB,R2       ; GET CPU LOWER BIAS
331              SWTCH1 #PCBIA         ; SET INPUT BUFFER
332              GETRV  R2,#2           ; READ CPU BIAS
333              SWTCH1          ; RESET BUFFER
334
335              MOV    $PCBIA,$RBLCK   ; SET UP LOWER BIAS
336              SWTCHO          ; RESET BUFFER
337              CALL   .ALOC2          ; ALLOCATE ICB
338              BCS    101$            ; IF ALLOCATION FAILURE - BRANCH
339              MOV    $PCBIA,$RBLCK   ; SET UP BIAS OF LOWER PARTITION
340
341              GETAD  R0,$ICBSZ       ; READ ICB
342              MOVB   $NVECT,$BFR     ; SET UP VECTOR
343              MOVB   $ICBLN,$BFR+1   ; SET UP ICB LENGTH
344              CLR    $BFR+2          ; CLEAR LINK WORD
345              PUTAD  $BFR+2          ; WRITE UPDATED ICB TO IMAGE
346
347              RESRG  <R1>            ; RESTORE KRB ADDRESS
348              GETRV  R1,#$KRBLN     ; READ KRB

```

VALT CREATED BY MACRO ON 8-JUL-85 AT 22:06 PAGE 3 J 11
 MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES									
ASL\$	#5-57									
CALL	6-159	6-170	6-182	6-192	6-213	6-221	9-379	9-453	11-527	11-550
	11-574	11-592	11-598	11-606	11-619	11-623	11-633			
EMSG\$R	#5-57	6-231	6-232	6-233	9-460	10-504	10-505	10-506	10-507	11-635
GETAD	#5-57	11-550	11-598	11-619						
GETRV	#5-58									
NKRDF\$	#5-57	5-69								
NTLER\$	#5-57	5-99	5-100	5-102	5-103	5-104	5-105	5-106	5-107	5-108
	5-109									
PUTAD	#5-57	11-623								
PUTRC	#5-57	9-453								
RAND	#9-453	9-453	#11-550	11-550	#11-598	11-598	#11-619	11-619	#11-623	11-623
RESRG	#5-58	9-454								
RETURN	6-222	9-456	10-499	11-626	12-662	13-685				
SAVRG	#5-58	9-377								
SWTCH1	#5-57	9-451	11-549	11-590	11-597	11-615	11-625			
SWTCHO	#5-57	9-452	11-591							

```

412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468

      .IF DF R$$MPL
      .DESPK - DEALLOCATE CORE BLOCK IN SECONDARY POOL
      THIS ROUTINE IS CALLED TO DEALLOCATE A SECONDARY POOL BLOCK. THE BLOCK
      IS INSERTED INTO THE FREE BLOCK CHAIN BY CORE ADDRESS. IF AN ADJACENT
      BLOCK IS CURRENTLY FREE, THEN THE TWO BLOCKS ARE MERGED.
      INPUTS:
      R0 = BIAS OF CORE BLOCK TO BE DEALLOCATED
      R1 = SIZE OF THE BLOCK IN UNITS OF 32 WORDS.
      OUTPUTS:
      THE CORE BLOCK IS MERGED INTO THE FREE CORE CHAIN BY ADDRESS
      AND MERGED WITH ANY BORDERING BLOCKS IF POSSIBLE
      :-
      .DESPK::SAVRG <R5,R4,R3,R2> ; SAV REG'S
      MOV #<1.LGTH+77>/100,R1 ; GET SIZE TO DEALLOCATE
      CLR -(SP) ; WORD FOR STORING TOP OF PREVIOUS BLOCK
      CLR R2 ; INITIAL PREV. BLOCK IN LIST IS LISTHEAD
      MOV $POLHD,R3 ; GET ADDRESS OF FIRST FREE BLOCK
      SWTCHI ; RESET INPUT BUFFER
      SWTCHO ; RESET OUTPUT BUFFER
      70$: TST R3 ; CHECK FOR END OF LIST
      BEQ 95$ ; IF END - BRANCH
      GETAD #140000,#4,R3 ; MAP NEXT FREE BLOCK
      CMP R0,R3 ; DOES BLOCK GO BEFORE CURRENT BLOCK
      BLO 80$ ; IF LO YES
      BEQ 150$ ; IF EQ, BLOCK OVERLAP ERROR
      MOV R3,(SP) ; GET ADDRESS OF CURRENT BLOCK
      ADD $BFR+2,(SP) ; CALCULATE ITS TOP ADDRESS
      MOV R3,R2 ; STORE ADDRESS AS THE PREVIOUS BLOCK
      MOV $BFR,R3 ; POINT TO NEXT BLOCK IN FREE LIST
      BR 70$
      80$: ADD R1,R0 ; GET TOP ADDRESS IN DEALLOCATED BLOCK
      CMP R0,R3 ; CAN IT BE COMBINED WITH THE NEXT BLOCK
      BNE 90$ ; IF NE NO
      MOV $BFR,-(SP) ; SAVE LINK TO NEXT FREE BLOCK
      MOV $BFR+2,-(SP) ; SAVE LENGTH
      SUB R1,R0 ; FIND START OF BLOCK BEING DEALLOCATED
      GETAD #140000,#4,R0 ; MAP IT
      MOV R1,$BFR+2 ; INSERT TO LENGTH OF THE BLOCK
      ADD (SP)+,$BFR+2 ; ADD LENGTH OF NEXT BLOCK
      MOV (SP)+,$BFR ; LINK TO NEXT FREE BLOCK
      PUTAD ; UPDATE IMAGE

```



```

255 000310 001005      BNE      60$      ; NO, CHECK HIGH SIDE BOUNDARY.
256 000312 060567 000002G      ADD      R5,$BFR+2      ; MERGE HOLES
257 000316      PUTAD      ; REWRITE BUFFER DESCRIPTOR
258 000322 000415      BR      70$      ; GO TEST FOR HIGH ADJANCY
259
260      ; LOW SIDE HDLE NOT CONTIGUDUS, POINT IT AT HOLE BEING RETURNED
261
262 000324 010367 000000G      60$:      MOV      R3,$BFR      ; WRITE BIAS OF HDLE BEING RETURNED
263 000330      PUTAD      ; TO LOW SIDE HOLE.
264 000334 016767 000000G 000000G      MOV      $BFR,$RBLCK      ; MAP TO CURRENT HOLE
265 000342 010067 000000G      MOV      R0,$BFR      ; LOAD POINTER TO NEXT HOLE
266 000346 010567 000002G      MOV      R5,$BFR+2      ; SIZE OF THIS HOLE
267 000352      PUTAD      ; AND REWRITE BUFFER
268
269      ; CHECK HIGH SIDE BOUNDARY CONDITION
270
271 000356 016746 000000G      70$:      MOV      $RBLCK,-(SP)      ; PUT BIAS OF CURRENT HOLE ON STACK
272 000362 066716 000002G      ADD      $BFR+2,(SP)      ; ADD CURRENT SIZE HOLE TO STACK
273 000366 026726 000000G      CMP      $BFR,(SP)+      ; AND CHECK FOR CONTIGUITY
274 000372 001023      BNE      80$      ; NOT CONTIGUDUS ...
275 000374 016700 000000G      MOV      $RBLCK,R0      ; SAVE CURRENT BIAS IN R0
276 000400 016767 000000G 000000G      MOV      $BFR,$RBLCK      ; MAP TO NEXT HOLE
277 000406      GETAD      ; READ IN NEXT DESCRIPTOR
278 000412 016746 000000G      MOV      $BFR,-(SP)      ; PUSH NEXT POINTER ON STACK
279 000416 016746 000002G      MOV      $BFR+2,-(SP)      ; PUSH NEXT HOLE SIZE ON STACK
280 000422 010067 000000G      MOV      R0,$RBLCK      ; REMAP TO LOW SIDE
281 000426      GETAD      ; REREAD LOW SIDE DESCRIPTOR
282 000432 062667 000002G      ADD      (SP)+,$BFR+2      ; ADD IN SIZE OF UPPER SIDE HOLE
283 000436 012667 000000G      MOV      (SP)+,$BFR      ; POINT AT NEXT HOLE
284 000442      80$:      PUTAD      ; REWRITE RECORD
285 000446      RETURN
286
287      000001      .END

```

VBIO CREATED BY MACRO ON 29-JUN-85 AT 02:12 PAGE 2 J 14
MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
ASR\$	#5-59 8-154
CALL	6-109 7-128
DIR\$	#5-59 9-187
GETAD	#5-59
GETRV	#5-59
PUTAD	#5-59 8-167
PUTRC	#5-59
QLOW\$	#5-59
RAND	#8-167 8-167
RETURN	6-113 7-133 8-172 9-192

```

LOCAL DATA
79
80
81
82
83
84
85 000000 000003
86 000002
87 000004 000000
88
89 000006
90 000012
91 000016

.SBTTL LOCAL DATA
:
: LOCAL DATA
:
XAVL: .WORD 3
      .BLKW 1
      .WORD 0
      : ROUNDING FACTOR
      : LISTHEAD
      : LENGTH OF THIS 'BLOCK'

LOCBUF: .BLKW 2
LOCBF1: .BLKW 2
LOCBF2: .BLKW 2
      : LOCAL BUFFER AREA
      : ANOTHER LOCAL BUFFER AREA
      : AND ANOTHER

```

VCEAL CREATED BY MACRO ON 29-JUN-85 AT 02:13 PAGE 1 J 16
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
DEAC	001162 R	11-355 #11-362
FE,CEX	= ***** GX	10-315 11-358
F.NRBD	= ***** GX	*7-122 *7-129 12-393 *7-152 12-408 *8-214 *8-231 *8-236 *8-249 *8-255 *8-259
		*14-497 *14-507
F.RSIZ	= ***** GX	*8-250 *8-257
F.URBD	= ***** GX	*7-121 *7-128 *7-151 *8-190 *8-204 *8-223 *8-260 *14-496 *14-506
LOCBF1	000012 R	#6-90 8-223 8-225
LOCBF2	000016 R	#6-91 8-200 8-202 8-203 8-204 8-206 *8-235 8-236 8-238
		*8-241 *8-244 8-249 *8-253 8-255
LOCBUF	000006 R	#6-89 *7-120 7-121 7-122 7-126 7-135 *7-138 7-140 7-144
		*7-147 8-190 *8-212 8-214 *8-228 *8-230 8-231 8-243 14-496
		14-497 14-499 *14-500 14-502 *14-504
N\$SVCT	= *****	7-133 7-145 8-188 8-203 14-495 14-502
R\$MPL	= *****	10-318 11-341 11-353
SYSFDB	= ***** GX	*7-121 *7-122 *7-128 *7-129 *7-151 *7-152 *8-190 *8-204 *8-214
		*8-223 *8-231 *8-236 *8-249 *8-250 *8-255 *8-257 *8-259 *8-260
		*14-496 *14-497 *14-506 *14-507
XAVL	000002 R	#6-86 7-115 7-150 7-153 8-193 8-254
Z.AVL	= ***** GX	*9-285 *9-292
\$ALLPR	001024 RG	#9-282
\$ALL15	001120 RG	7-130 10-316 #10-324
\$ALL16	001076 RG	#10-314
\$ALL18	001232 RG	#12-407
\$ALL22	001232 RG	#12-406
\$ALN18	001170 RG	#12-392
\$ALPOL	000022 RG	#7-113
\$BFR	= ***** GX	7-128 7-129 7-151 7-152 8-259 8-260 14-506 14-507
\$CEACC	= ***** GX	7-133 7-145 8-188 8-203 8-222 14-495 14-502
\$CEAVL	= ***** GX	10-319 11-356
\$CRAVL	= ***** GX	11-351
\$DEA16	001126 RG	8-184 #11-348
\$DEA18	001306 RG	#13-446
\$DEA22	001306 RG	#13-445
\$DEPOL	000274 RG	#8-181
\$XSIZ	= ***** GX	11-354
\$Fmask	= ***** GX	10-315 11-358
\$GETAD	= ***** GX	7-134 7-146 12-393 12-408 13-447
\$NBias	= ***** GX	12-400 *12-403 8-191 8-205 8-224 14-498 14-503
\$PDVA	= ***** GX	9-285 9-291 *13-461
\$PRAVL	= ***** GX	9-284 9-292
\$PUTAD	= ***** GX	7-139 7-148 8-215 8-232 8-237 8-250 8-257 14-501 14-505
\$OBias	= ***** GX	12-415 *12-418 13-454 *13-463
\$QSTRT	= ***** GX	13-455 13-459
\$RADDR	= ***** GX	*7-134 *7-139 *7-146 *7-148 *8-191 *8-205 *8-215 *8-224 *8-232
		*8-237 *8-250 *8-257 *14-498 *14-501 *14-503 *14-505 *8-242 *8-248
\$RBLCK	= ***** GX	8-192 8-211 *8-213 *8-216 8-221 *8-229 *8-233
		*8-252
\$RSIZE	= ***** GX	*7-124 *8-189 *14-493
\$SAVAL	= ***** GX	11-349 13-450
\$SAVRG	= ***** GX	12-396 12-411
\$XAVL	= ***** GX	7-118 *7-150 *7-153 8-198 *8-254
\$XLINX	001374 RG	#14-492

```

197
198
199
200
201
202
203
204
205
206
207
208 000140
209 000140
210 000144 012700 000000G
211 000150 012701 000000G
212 000154 005002
213 000156
214

;+
; $INT1 - ".INT1" ACTION ROUTINE
;
; INPUTS:
;     NONE
;
; OUTPUTS:
;     R0,R1,R2=DESTROYED
;     $VECT1=CURRENT COPY OF $SIZE
;-

$INT1::
    CALL    $VFDDM          ; MAKE SURE IT'S A DDM
    MOV     #F1.IN1,R0      ; BIT TO BE SET
    MOV     #F1.INT,R1      ; BIT WHICH MUST NOT BE SET
    CLR     R2              ; VECTOR INDEX
    CALLR   VFVCT           ; CHECK VECTOR AVAILABILITY

```

A\$\$CHK= 000000	F2.FILE= ***** GX	P\$\$WRD= 000000	\$ERR43 000315R	002 \$RQ1 = ***** GX
A\$\$CPS= 000000	G\$\$TTP= 000000	Q\$\$OPT= 000010	\$ERR81 000416R	002 \$RQ2 = ***** GX
A\$\$PRI= 000000	G\$\$TSS= 000000	REP2B 001420RG	\$ERR85 000444R	002 \$RQ3 = ***** GX
A\$\$TRP= 000000	G\$\$TTK= 000000	R\$\$DER= 000000	\$EVEN = ***** GX	\$RQ4 = ***** GX
CLERR 000770R	G\$\$WRD= 000000	R\$\$K11= 000001	\$FILE 000000RG	\$SECSR 001164RG
C\$\$CKP= 000000	I\$\$RAR= 000000	R\$\$SND= 000000	\$FILEN= ***** GX	\$SIZE = ***** GX
C\$\$ORE= 000400	I\$\$RDN= 000000	R\$\$11M= 000000	\$FILEX 000050RG	\$TERR = ***** GX
C\$\$RSH= 177564	K\$\$CNT= 177546	SYSFDB= ***** GX	\$FLAGS= ***** GX	\$TMFIX= ***** GX
D\$\$BUG= 177514	K\$\$CSR= 177546	S\$\$WRG= 000000	\$FLAG1= ***** GX	\$VALUE= ***** GX
D\$\$ISK= 000000	K\$\$LDC= 000000	S\$\$YSZ= 007600	\$FLAG2= ***** GX	\$VECT1= ***** GX
D\$\$L11= 000001	K\$\$TPS= 000074	TMPEXT 000000R	002 \$GETRV= ***** GX	\$VFDOM= ***** GX
D\$\$YNC= 000000	LD\$LP = 000000	T\$\$KMG= 000000	\$INT 000116RG	\$VFDLC= ***** GX
D\$\$YNM= 000000	L\$\$ASG= 000000	T\$\$MIN= 000000	\$INTX 000064RG	\$VFMUX 001234RG
E\$\$XPR= 000000	L\$\$DRV= 000000	V\$VCT= 001254RG	\$INT1 000140RG	\$VFSPC= ***** GX
FL.CHA= ***** GX	L\$\$P11= 000001	V\$\$CTR= 001000	\$INT2 000162RG	\$MTP = ***** GX
FL.LMC= ***** GX	L\$\$11R= 000000	X\$\$DBT= 000000	\$INT3 000206RG	\$MUCT = ***** GX
FL.MUX= ***** GX	MAX14 000502R	\$ALL16= ***** GX	\$LFILE 000602RG	\$VECT= ***** GX
FMT2 000646R	002 M\$\$CRB= 000124	\$AZFS = ***** GX	\$LFLD = ***** GX	.EOF = ***** GX
FMT2B 000665R	002 M\$\$CRX= 000000	\$BFR = ***** GX	\$MPLD = ***** GX	.FILE = ***** GX
FMT3 000622R	002 M\$\$FCS= 000000	\$CBOMG= ***** GX	\$MPTB 000534RG	.INT = ***** GX
FM.2 = 000000	M\$\$MGE= 000000	\$CFILE 001050RG	\$MXADD= ***** GX	.LFILE= ***** GX
FM.2B = 000000	M\$\$NET= 000000	\$CL10S= ***** GX	\$MXLEN= ***** GX	.MPTAB= ***** GX
FM.3 = 000000	M\$\$DVR= 000000	\$CLQUE= ***** GX	\$MXPTB 000232RG	.MXPTB= ***** GX
F\$SLVL= 000001	NSITAB 000004R	002 \$ELINE= ***** GX	\$MXSIZ= ***** GX	.MXTAB= ***** GX
F.RSIZ= ***** GX	N\$\$ACC= 000001	\$ERRKX 000474R	002 \$MXTAB 000776RG	.NS0 = ***** GX
F1.INT= ***** GX	N\$\$BUF= 000001	\$ERRKY 000542R	002 \$MXVCT= ***** GX	.NS1 = ***** GX
F1.IN1= ***** GX	N\$\$LDV= 000001	\$ERR17 000024R	002 \$NALL = ***** GX	.NS2 = ***** GX
F1.IN2= ***** GX	N\$\$MCP= 000001	\$ERR22 000360R	002 \$NLIB = ***** GX	.NS3 = ***** GX
F1.IN3= ***** GX	N\$\$MLL= 000001	\$ERR23 000054R	002 \$NVECT= ***** GX	.NS4 = ***** GX
F1.LFI= ***** GX	N\$\$MOV= 000010	\$ERR24 000126R	002 \$OFF = ***** GX	.NS5 = ***** GX
F1.LFL= ***** GX	N\$\$NCT= 000001	\$ERR25 000156R	002 \$PRV.N= ***** GX	.NS6 = ***** GX
F1.MPL= ***** GX	N\$\$PEM= 000001	\$ERR41 000206R	002 \$RADDR= ***** GX	.NS7 = ***** GX
F1.MPT= ***** GX	P\$\$P45= 000000	\$ERR42 000252R	002 \$REP.C= ***** GX	.SECSR= ***** GX
F1.MXT= ***** GX				

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
 001454 001 (RW,I,LCL,REL,CON)
 DATA 000716 002 (RW,N,LCL,REL,CON)
 Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 10422 Words (41 Pages)
 Size of core pool: 14440 Words (55 Pages)
 Operating system: RSX-11M/PLUS

Elapsed time: 00:00:21.04
 SY:VAC2.V2,[132,134]VAC2/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VAC2

VAC3 CREATED BY MACRO ON 29-JUN-85 AT 02:09 PAGE 1 K 3
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
ASYM	000000 R	#5-73 6-132 6-147 7-192
FMT2A	000232 R	5-94 #5-104
FMT4	000262 R	5-95 5-96 5-97 5-98 #5-105
FM.2A	= 000000	#5-94
FM.4	= 000000	#5-95 #5-96 #5-97 #5-98
F1. STE	= ***** GX	6-129 6-171
GBLK	000006 R	#5-78 6-158
GETAB	000022 R	#5-86 6-172
GSYM	000012 R	#5-79 *6-150 *6-153
GVAL	000016 R	#5-80 6-161
REP2A	000210 R	5-94 #7-187
STBEXT	000312 R	#5-110 6-156
SS\$BAS	= *****	5-94 5-95 5-96 5-97 5-98
\$AZFS	= ***** GX	6-157
\$CAT5	= ***** GX	6-149 6-152
\$CLEVL	= ***** GX	6-169
\$ELINE	= ***** GX	5-94
\$ERR27	000032 R	5-85 #5-94
\$ERR29	000054 R	5-86 #5-95
\$ERR30	000114 R	5-87 #5-96
\$ERR31	000154 R	5-88 #5-97
\$ERR32	000204 R	5-89 #5-98
\$FLAG1	= ***** GX	6-129 *6-171
\$GLOBL	000000 RG	#6-127
\$PRV.N	= ***** GX	5-95 5-96
\$REP.C	= ***** GX	7-187
\$REP.P	= ***** GX	5-97 5-98
\$SYMBL	= ***** GX	6-159
\$TERR	= ***** GX	5-95 5-96 5-97 5-98
\$TWARN	= ***** GX	5-94
\$VALUE	= ***** GX	*6-128 *6-161
.PSTCN	= ***** GX	6-133
.PSTPT	= ***** GX	6-131

```

400                                     ;+
401                                     ; REP6 - REPLACE PROCESS NAME AND THEN FORM APR NUMBER
402                                     ;
403                                     ; INPUTS:
404                                     ;   R3=FORMAT STRING ADDRESS
405                                     ;   R4=ERROR MESSAGE BUFFER ADDRESS
406                                     ;   $CAPR=APR NUMBER IN BINARY
407                                     ;
408                                     ; OUTPUTS:
409                                     ;   R0=DESTROYED
410                                     ;   R3,R4=UPDATED
411                                     ; -
412
413 001100 REP6: CALL $REP,C           ; REPLACE PROCESS NAME
414 001104 010500 MOV R5,R0          ; COPY ERROR MESSAGE BLOCK ADDRESS
415 001106 005720 TST (R0)+          ; NOW POINTS AT TEXT STRING
416 001110 122720 000120 10$: CMPB #'P',(R0)+ ; LOOK FOR 'P'
417 001114 001375 BNE 10$           ;
418 001116 122720 000122 CMPB #'R',(R0)+ ; FOLLOWED BY 'R'
419 001122 001372 BNE 10$           ;
420 001124 112710 000060 MOVB #'0',(R0) ; FORM APR NUMBER
421 001130 1567'0 000000G BISB $CAPR,(R0) ;
422 001134 RETURN                   ;

```


A\$CHK= 000000	G\$STPP= 000000	M\$SCRB= 000124	S\$SWRG= 000000	\$ERR71 000040R	002
A\$CPS= 000000	G\$STSS= 000000	M\$SCRX= 000000	S\$SYSZ= 007600	\$ERR72 000070R	002
A\$PRI= 000000	G\$STTK= 000000	M\$SFCS= 000000	T\$SNHD= ***** GX	\$ERR73 000144R	002
A\$TRP= 000000	G\$SWRD= 000000	M\$SMGE= 000000	T\$KMG= 000000	\$ERR74 000206R	002
BN.LBL 000204R	I\$RAR= 000000	M\$SNET= 000000	T\$SMIN= 000000	\$ERR75 000260R	002
BN.OPN 000130R	I\$RDN= 000000	M\$SOVR= 000000	V\$CTR= 001000	\$FILEN= ***** GX	
C\$CKP= 000000	K\$CNT= 177546	N\$ACC= 000001	X\$DBT= 000000	\$FLAG= ***** GX	
C\$ORE= 000400	K\$CSR= 177546	N\$BUF= 000001	\$ABFI 000000RG	\$FLAG2= ***** GX	
C\$RSH= 177564	K\$LDC= 000000	N\$LDV= 000001	\$AZFS = ***** GX	\$IOSB = ***** GX	
DSKBUF 000000R	K\$TPS= 000074	N\$MCP= 000001	\$BFALL= ***** GX	\$LBN = ***** GX	
D\$BUG= 177514	LD\$LP = 000000	N\$MLL= 000001	\$FBAS= ***** GX	\$PRV.N= ***** GX	
D\$ISK= 000000	L\$EXT= ***** GX	N\$MOV= 000010	\$FBFB= ***** GX	\$REP.C= ***** GX	
D\$LI1= 000001	L\$FLG= ***** GX	N\$NCT= 000001	\$FBFB= ***** GX	\$REP.P= ***** GX	
D\$SYNC= 000000	L\$BLDZ= ***** GX	N\$PEM= 000001	\$BFXFR= ***** GX	\$RLBL = ***** GX	
D\$SYM= 000000	L\$BMXZ= ***** GX	P\$P45= 000000	\$CAPR = ***** GX	\$RQ5 = ***** GX	
E\$XPR= 000000	L\$BSA = ***** GX	P\$WRD= 000000	\$CLDEQ= ***** GX	\$RQ7 = ***** GX	
FL.CHA= ***** GX	L\$ASG= 000000	Q\$OPT= 000010	\$CLIOS= ***** GX	\$TERR = ***** GX	
FM16 000332R	L\$DRV= 000000	R\$DER= 000000	\$CLOPE= ***** GX	\$VALUE= ***** GX	
FM.6 = 000000	L\$P11= 000001	R\$K11= 000001	\$CUR.N= ***** GX	\$VFLV1= ***** GX	
F\$LVL= 000001	L\$11R= 000000	R\$SND= 000000	\$ELINE= ***** GX	\$BASEB= ***** GX	
F2.FIL= ***** GX	MAPBYT 000354R	002 R\$11M= 000000	\$ERR69 000000R	002 .MGME= ***** GX	

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
 . 000412 001 (RW,I,LCL,REL,CON)
 DATA 000356 002 (RW,D,LCL,REL,CON)
 .BUF 000002 003 (RW,I,LCL,REL,CON)
 Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 10306 Words (41 Pages)
 Size of core pool: 14440 Words (55 Pages)
 Operating system: RSX-11M/PLUS

Elapsed time: 00:00:11.09

SY:VAC5.V2,[132,134]VAC5/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VAC5

```

250                                     ;+
251                                     ; $LINKS - ".LINKS" ACTION ROUTINE
252                                     ;
253                                     ; INPUTS:
254                                     ;     NONE
255                                     ;
256                                     ; OUTPUTS:
257                                     ;     R2=DESTROYED
258                                     ; -
259 $LINKS::
260     CALL    $VFLLC                ; MAKE SURE IT'S AN LLC
261     CALL    $RQT                  ; REQUEST BINARY BUFFER ACCESS
262     BCS     10$                  ; IF CS, NO ACCESS
263     MOVB    #.LINKS, (R2)        ; STORE FUNCTION CODE
264     ADD     #2, $SIZE            ; ADJUST DATA BASE SIZE
265     RETURN
10$:

```

```

56      ;***
57      ; LIBRARY MACROS
58      ;***
59      .MCALL  MSG$,MSG$,NTLRS$
60      .ENABL  LC
61
62      ;***
63      ; LOCAL DATA
64      ;***
65
66      000000      .PSECT  DATA,D
67
68      ;
69      ; ALLOCATED LIBRARY BLOCK ADDRESS
70      ;
71      000000      LBLK:: .BLKW  1
72
73      ;
74      ; ERROR MESSAGES
75      ;
76      000002      NTLRS$ ,Y2,6,$TWARN,REP6,$ELINE,<not built for APR*>
77      000036      NTLRS$ ,Y7,6,$TWARN,$REP.C,$ELINE,<Base address not zero.>
78      000076      NTLRS$ ,Y3,6,$TERR,$REP.P,$ELINE,<Base address not a multiple of 4k.>
79      000152      NTLRS$ ,Y4,2,$TERR,$REP.C,$ELINE,<Resource allocation failure.>
80      000220      NTLRS$ ,Y6,2,$TERR,$REP.C,$ELINE,<Maximum allocations/libraries is 14.>
81
82      ;
83      ; ERROR MESSAGE FORMAT STRINGS
84      ;
85
86      000276      052      040      111      FMT6:  .ASCIZ  '* Image file -- '
87      000301      155      141      147
88      000304      145      040      146
89      000307      151      154      145
90      000312      040      055      055
91      000315      040      000
92      000317      052      040      124      FMT2:  .ASCIZ  '* Template -- '
93      000322      145      155      160
94      000325      154      141      164
95      000330      145      040      055
96      000333      055      040      000
97
98      .EVEN
99      .GLOBL  FMT2,FMT6
100
101      .PSECT

```


ALVLIB - ALLOCATE VECTOR BLOCK FOR LIBRARY

```

193      .SBTTL  ALVLIB - ALLOCATE VECTOR BLOCK FOR LIBRARY
194
195      *
196      ALVLIB - ALLOCATE VECTOR BLOCK
197
198      INPUTS:
199      $LBVT - ADDRESS OF VECTOR TABLE
200      $LBVL - SIZE OF VECTOR TABLE (WORDS)
201      BFADR - BIAS OF LIBRARY
202
203      OUTPUTS:
204      VECTOR BLOCK ALLOCATED AND INITIALIZED
205
206      .IF DF R$$MPL
207
208      ALVLIB: SAVRG <R1>          ; SAVE PCB ADDRESS
209      MOV      #V.LEN,R1         ; GET SIZE OF VECTOR BLOCK
210      CALL     $ALPOL            ; ALLOCATE FROM BYTE POOL
211      MOV      $NTLHB+.CXVEC,$BFR+V.LNK ; GET ADDRESS OF OLD FIRST BLOCK
212      MOV      R0,$NTLHB+.CXVEC   ; LINK IN NEW VECTOR BLOCK
213
214      MOV      $LBVT,$BFR+V.ADD   ; COPY TABLE ADDRESS
215      MOV      $LBVL,$BFR+V.SIZ   ; COPY TABLE SIZE
216      MOV      BFADR,$BFR+V.BIAS  ; STORE LIBRARY BIAS
217      MOV      #VF.LIB,$BFR+V.FLG ; SET LIBRARY VECTOR BLOCK FLAG
218
219      CEACCS$ R0,R2              ; MAP TO VECTOR BLOCK
220      PUTAD    R2,#V.LEN         ; UPDATE IMAGE
221      RESRG    <R1>              ; RESTORE PCB ADDRESS
222
223      RETURN
224
225      .ENDC
226
227      .END

```

000001

VALT - VNP LINE TABLE ALLOCATIO MACRO V05.03b Monday 08-Jul-85 22:06 ^{K 10} Page 8-1
\$ALICB - ALLOCATE ICB

```
349          ASR      R0          ; DIVIDE ICB ADDRESS BY 2
350          ADD      CPU,R0      ; FORM LINK POINTER
351          MOV      R0,$BFR     ; UPDATE KRB
352          PUTRC     ; WRITE UPDATED KRB TO IMAGE
353
354          RESRG     <R3>       ; RESTORE REG
355          RESRG     <R2>       ; RESTORE REG
356
357          RETURN
358
359          101$:      MSG$R 9D    ; ICB ALLOCATION FAILURE
360
361          .ENDC
```

```

LL          SSSSSSSS  TTTTTTTTTT
LL          SSSSSSSS  TTTTTTTTTT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SSSSSS    TT
LL          SSSSSS    TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LLLLLLLLLL SSSSSSSS  TT
LLLLLLLLLL SSSSSSSS  TT

```

```

469          BR      100$
470
471          90$: BHI    150$      ; IF HI, DEALLOCATED BLOCK OVERLAPS NEXT
472          SUB    R1,R0      ; GET BASE ADDRESS OF DEALLOCATED BLOCK
473          95$: GETAD #140000,#4,R0 ; MAP BLOCK BEING DEALLOCATED
474          MOV    R3,$BFR    ; INSERT LINK TO NEXT BLOCK IN FREE LIST
475          MOV    R1,$BFR+2  ; INSERT LENGTH
476          PUTAD   ; UPDATE IMAGE
477
478          100$: CMP    (SP)+,R0 ; CAN BLOCK BE MERGED WITH PREVIOUS BLOCK
479          BNE    110$      ; IF NE NO
480          MOV    $BFR,-(SP)  ; SAVE LINK TO NEXT FREE BLOCK
481          MOV    $BFR+2,-(SP) ; AND LENGTH
482          GETAD #140000,#4,R2 ; MAP PREVIOUS BLOCK IN FREE LIST
483          ADD    (SP)+,$BFR+2 ; ADD LENGTH TO WHAT IS ALREADY THERE
484          MOV    (SP)+,$BFR  ; LINK TO NEXT BLOCK
485          PUTAD   ; UPDATE IMAGE
486          BR      130$
487
488          110$: BHI    150$      ; IF HI, BLOCK OVERLAPS PREVIOUS BLOCK
489          TST    R2          ; CHECK FOR END OF LIST
490          BNE    120$      ; IF END - BRANCH
491          GETAD #140000,#4,R2 ; MAP PREVIOUS BLOCK
492          MOV    R0,$POLHD  ; USE THE LIST HEAD AS THE PREVIOUS BLOCK
493          BR      130$      ; CONTINUE
494          120$: MOV    R0,$BFR ; LINK BLOCK BEING DEALLOCATED INTO CHAIN
495          PUTAD   ; UPDATE IMAGE
496
497          130$: SWTCHI #SECTMP ; SET BUFFER ADDRESS
498          SWTCHO #SECTMP      ; ...
499          GETRV  .SECFR,#2    ; READ $SECFR VALUE
500
501          ADD    R1,SECTMP    ; ADD AMOUNT DEALLOCATED TO TOTAL FREE
502          PUTRC  .SECFR      ; UPDATE $SECFR
503          SWTCHI .SECFR      ; RESET BUFFER
504          SWTCHO .SECFR      ; ...
505
506          RESRG  <R2,R3,R4,R5> ; RESTORE REG'S
507
508          140$: RETURN
509
510          150$: CRASH      ; BAD DEALLOCATION
511
512          .ENDC
513
514          000001          .END
515

```


A\$\$CHK= 000000	F\$\$LVL= 000001	L\$\$ASG= 000000	N\$\$MCP= 000001	S\$\$WRG= 000000
A\$\$CPS= 000000	F.RSIZ= ***** GX	L\$\$DRV= 000000	N\$\$MLL= 000001	S\$\$YSZ= 007600
A\$\$PRI= 000000	G\$\$TPP= 000000	L\$\$P11= 000001	N\$\$MOV= 000010	T\$\$KMG= 000000
A\$\$TRP= 000000	G\$\$TSS= 000000	L\$\$11R= 000000	N\$\$NCT= 000001	T\$\$MIN= 000000
C\$\$CKP= 000000	G\$\$TTK= 000000	M\$\$CRB= 000124	N\$\$PEM= 000001	V\$\$CTR= 001000
C\$\$ORE= 000400	G\$\$WRD= 000000	M\$\$CRX= 000000	P\$\$P45= 000000	X\$\$DBT= 000000
C\$\$RSH= 177564	I\$\$RAR= 000000	M\$\$FCS= 000000	P\$\$WRD= 000000	\$BFR = ***** GX
D\$\$BUG= 177514	I\$\$RDN= 000000	M\$\$MGE= 000000	Q\$\$OPT= 000010	\$GETAD= ***** GX
D\$\$ISK= 000000	K\$\$CNT= 177546	M\$\$NET= 000000	R\$\$DER= 000000	\$PUTAD= ***** GX
D\$\$L11= 000001	K\$\$CSR= 177546	M\$\$QVR= 000000	R\$\$K11= 000001	\$RADDR= ***** GX
D\$\$YNC= 000000	K\$\$LDC= 000000	N\$\$ACC= 000001	R\$\$SND= 000000	\$RBLCK= ***** GX
D\$\$YNM= 000000	K\$\$TPS= 000074	N\$\$BUF= 000001	R\$\$11M= 000000	.AMEM 000000RG
E\$\$XPR= 000000	LD\$L P = 000000	N\$\$LDV= 000001	SYSFDB= ***** GX	.DMEM 000172RG

. ABS. 000000 000 (RW,I,GBL,ABS,QVR)
 000450 001 (RW,I,LCL,REL,CON)
 Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 9589 Words (38 Pages)
 Size of core pool: 14440 Words (55 Pages)
 Operating system: RSX-11M/PLUS

Elapsed time: 00:00:10.43
 SY:VAL22.V2,[132,134]VAL22/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VAL22


```

93      .SBTTL $ALPOL - ALLOCATE FROM EXTENDED POOL
94
95      +
96      *** - $ALPOL - ALLOCATE SINGLE-WORD STORAGE FROM THE NETWORK POOL
97
98      THIS ROUTINE IS CALLED TO ALLOCATE AN CORE BUFFER IN EXTENDED POOL.
99      THE ALLOCATION ALGORITHM IS FIRST FIT AND BLOCKS ARE ALLOCATED IN
100     MULTIPLES OF FOUR BYTES. IF THE EXTENDED ALLOCATION FAILS, THE
101     BLOCK IS ALLOCATED USING RSX POOL.
102
103     INPUTS:
104     R1 = LENGTH OF ALLOCATION IN BYTES
105     $XBIAS = BIAS OF START OF EXTENDED POOL
106
107     OUTPUTS:
108     C-BIT = SUCCESS/FAILURE
109     R0 = UNMAPPED ADDRESS OF ALLOCATION
110     R1 = LENGTH OF ALLOCATION IN BYTES
111
112     -
113
114     $ALPOL::
115     MOV     R3,-(SP)      ; SAVE REGISTER
116     MOV     #XAVL-2,R0   ; POINT TO ALLOCATION MASK WORD
117     ADD     (R0),R1      ; ADD ROUNDING FACTOR
118     BIC     (R0)+,R1     ; CLEAR EXCESS AND POINT TO LIST HEAD
119     MOV     $XAVL,(R0)   ; COPY CEX POOL LISTHEAD
120     MOV     R0,R2        ; INITIALIZE CURRENT UNMAPPED POINTER (R2)
121     MOV     (R0),LOCBUF  ; COPY CEX POLL LISTHEAD AGAIN
122     SWTCHI  #LOCBUF      ; USE LOCBUF FOR INPUT BUFFER
123     SWTCHO  #LOCBUF      ; AND OUTPUT BUFFER
124
125     000062 012767 000004 000000G 10$: MOV     #4,$RSIZE      ; SET UP FOR 4 BYTE READS
126     000070 010203          177710  MOV     R2,R3          ; MOVE CURRENT UNMAPPED PTR TO PREV UNM PTR
127     000072 016702          177710  MOV     LOCBUF,R2      ; GET UNMAPPED ADDRESS OF NEXT BLOCK (CURRENT)
128     000076 001011          177710  BNE     20$            ; IF NE, ANOTHER BLOCK TO CHECK
129     000100          177710  SWTCHI  ; RESTORE DEFAULT INPUT
130     000106          177710  SWTCHO  ; AND OUTPUT BUFFERS
131     000114          177710  CALL    $ALL15                ; ELSE, TRY TO ALLOCATE FROM RSX POOL
132     000120 000452          177710  BR     40$            ; AND RETURN
133
134     20$: CEACCS R2,R0      ; CONVERT TO MAPPED ADDRESS
135     GETAD  R0            ; READ IN FOUR BYTES
136     CMP    LOCBUF+2,R1   ; BLOCK BIG ENOUGH?
137     BLQ    10$          ; IF LO NO
138     BEQ    30$          ; IF EQ BLOCK IS EXACT SIZE
139     SUB    R1,LOCPUF+2   ; CALCULATE SIZE REMAINING
140     PUTAD  R0            ; WRITE OUT BLOCK WITH UPDATED SIZE
141     ADD    LOCBUF+2,R2   ; ADD SIZE OF REMAINING BLOCK TO UNM ADDR
142     MOV    R2,R0        ; MOVE RETURNED ADDRESS TO R0
143     BR     40$          ; AND RETURN
144
145     30$: MOV     LOCBUF,-(SP) ; SAVE NEXT BLOCK ADDRESS
146     CEACCS R3,R0          ; CONVERT TO MAPPED ADDRESS
147     GETAD  R0            ; READ FOUR BYTE LISTHEAD
148     MOV    (SP)+,LOCBUF  ; UNLINK BLOCK FROM LIST
149     PUTAD  R0            ; WRITE BLOCK BACK TO FILE
150     MOV    R2,R0        ; AND RETURN UNMAPPED ADDRESS TO USER

```

VCEAL CREATED BY MACRO ON 29-JUN-85 AT 02:13 PAGE 2 K 16

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
.ALLPR	= ***** GX	9-286
.ALOCB	= ***** GX	10-325
.ALOC1	= ***** GX	10-320
.AMEM	= ***** GX	12-401 12-416
.DEAC1	= ***** GX	11-362
.DMEM	= ***** GX	13-458

```

216
217      ;+ $INT2 - ".INT2" ACTION ROUTINE
218      ;
219      ; INPUTS:
220      ;     NONE
221      ;
222      ; OUTPUTS:
223      ;     R0,R1,R2=DESTROYED
224      ;     $VECT2=CURRENT COPY OF $SIZE
225      ; -
226
227      $INT2::
228      CALL    $VFDDM          ; MAKE SURE IT'S A DDM
229      MOV     #F1.INT2,R0     ; BIT TO BE SET
230      MOV     #F1.INT,R1     ; BIT WHICH MUST NOT BE SET
231      MOV     #2,R2          ; VECTOR INDEX
232      CALLR   VFVCT          ; CHECK VECTOR AVAILABILITY
233
227 000162
228 000162
229 000166 012700 000000G
230 000172 012701 000000G
231 000176 012702 000002
232 000202
233

```

VAC2 CREATED BY MACRO ON 29-JUN-85 AT 02:09 PAGE 1 L 2
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
CTLERR	000770 R	14-385 #14-415 15-448
FLCHA	= ***** GX	12-300 18-538
FLLMC	= ***** GX	16-484
FLMUX	= ***** GX	17-509
FMT2	000646 R	5-94 5-98 5-99 5-101 5-103 5-105 5-106 5-107 5-108
FMT2B	000665 R	#5-113 5-95 5-96 #5-115
FMT3	000622 R	5-92 #5-112
FM.2	= 000000	#5-94 #5-98 #5-99 #5-101 #5-103 #5-105 #5-106 #5-107 #5-108
FM.2B	= 000000	#5-95 #5-96
FM.3	= 000000	#5-92
F.RSIZ	= ***** GX	*18-544
F1.INT	= ***** GX	8-191 9-211 10-230 11-249
F1.IN1	= ***** GX	8-192 9-210
F1.IN2	= ***** GX	8-192 10-229
F1.IN3	= ***** GX	8-192 11-248
F1.LFI	= ***** GX	14-377
F1.LFL	= ***** GX	14-380
F1.MPL	= ***** GX	14-370
F1.MPT	= ***** GX	14-367
F1.MXT	= ***** GX	12-272 15-435
F2.FIL	= ***** GX	6-136
IS\$AS	= *****	18-542
MAX14	000502 R	12-280 #13-337
NSITAB	000004 R	#5-80 18-545
REP2B	001420 RG	5-95 5-96 #19-571
R\$SEIS	= *****	18-542
R\$311D	= *****	18-542
SYSFDB	= ***** GX	*18-544
S\$BAS	= *****	5-92 5-94 5-94 5-94 5-95 5-95 5-96 5-96 5-98
		5-98 5-99 5-99 5-101 5-101 5-103 5-103 5-105 5-105
		5-106 5-106 5-107 5-107 5-108 5-108
TMPEXT	000000 R	#5-72 6-140
VFCVT	001254 RG	8-194 9-213 10-232 11-251 #18-528
\$ALL16	= ***** GX	12-305
\$AZFS	= ***** GX	6-141
\$BFR	= ***** GX	18-545
\$CBOMG	= ***** GX	19-580
\$CFILE	001050 RG	#15-445
\$CLIOS	= ***** GX	6-145
\$CLOUE	= ***** GX	6-142
\$ELINE	= ***** GX	5-92 5-94 5-95 5-96 5-98 5-99 5-101 5-103 5-105
		5-106 5-107 5-108
\$ERRKX	000474 R	#5-107 12-321
\$ERRKY	000542 R	#5-108 13-348
\$ERR17	000024 R	#5-92 6-154
\$ERR22	000360 R	#5-103 12-320 14-415
\$ERR23	000064 R	#5-94 18-553
\$ERR24	000126 R	#5-95 18-554
\$ERR25	000156 R	#5-96 18-555
\$ERR41	000206 R	#5-98 14-413
\$ERR42	000252 R	#5-99 14-414

VAC3 CREATED BY MACRO ON 29-JUN-85 AT 02:09 PAGE 2 L 3

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

CALL	6-149	6-152	6-157	6-159	7-187	
CALLR	6-173					
NTLR\$	#5-59	5-94	5-95	5-96	5-97	5-98
RETURN	6-164	7-201				

```

424      *
425      ZER18  - ZERO 18-BIT ALLOCATION
426
427      INPUTS:
428      OUTPUT FROM $ALL18 OR ALN18
429
430      OUTPUTS:
431      ALLOCATION IS ZEROED
432
433      ZER16  - ZERO 16-BIT ALLOCATION
434
435      INPUTS:
436      OUTPUT FROM $ALL15 OR $ALL16
437
438      OUTPUT:
439      ALLOCATION IS ZEROED
440
441      -
442
443      .ENABL LSB
444
445      001136      ZER18:  MOV    R0,$RBLCK      ; SAVE BLOCK NUMBER ( BIAS )
446      001136      010067 000000G      CLR    $RADDR      ; NO VIRTUAL ADDRESS
447      001142      005067 000000G      MOV    R1,-(SP)      ; SAVE R1
448      001146      010146      ASL$      6,R1      ; COMPUTE NUMBER OF BYTES
449      001150      MOV    R1,$RSIZE      ; COPY NUMBER OF BYTES
450      001164      010167 000000G      MOV    (SP)+,R1      ; RESTORE R1
451      001170      012601
452
453      001172      005767 000000G      TST    .MGMGE      ; MEMORY MANAGEMENT SYSTEM ?
454      001176      002006      BGE    811$      ; IF GE, YES
455
456      001200      010067 000000G      ZER16:  MOV    R0,$RADDR      ; SET ADDRESS
457      001204      005067 000000G      CLR    $RBLCK      ; AND BIAS
458      001210      010167 000000G      MOV    R1,$RSIZE      ; AND SIZE
459
460      001214      010046      811$:  MOV    R0,-(SP)      ; SAVE BLOCK COUNT
461      001216      010146      MOV    R1,-(SP)      ; SAVE NUMBER OF BLOCKS
462      001220      012700      MOV    #512,R0      ; PREPARE TO ZERO 256. WORDS
463      001224      012701      MOV    #256.,R1      ;
464      001230      005020      82$:  CLR    (R0)+      ; ZERO THEM
465      001232      005301      DEC    R1
466      001234      001375      BNE    82$
467      001236      016701      MOV    $RSIZE,R1      ; SET SIZE OF TRANSFER
468      001242      012767 001000 000000G      MOV    #1000,$RSIZE      ; SET MAXIMUM TRANSFER SIZE
469      001250      026701 000000G      CMP    $RSIZE,R1      ; LIMIT TRANSFERS TO 512. BYTES
470      001254      003402      BLE    83$
471      001256      010167 000000G      822$:  MOV    R1,$RSIZE      ; SET LAST TRANSFER SIZE
472      001262      001407      83$:  PUTAD  $RSIZE,R1      ; WRITE ZEROS
473      001266      166701 000000G      SUB    $RSIZE,R1      ; SUBTRACT OUT SIZE WRITTEN
474      001272      001407      BEQ    84$      ; ALL DONE
475      001274      066767 000000G 000000G      ADD    $RSIZE,$RADDR      ; SET NEW DESTINATION ADDRESS
476      001302      026701 000000G      CMP    $RSIZE,R1      ; IS THIS A SHORT BLOCK ?
477      001306      003765      BLE    83$      ; NO
478      001310      000762      BR     822$      ; YES
479      001312      012601      84$:  MOV    (SP)+,R1      ; RESTORE REGISTERS
480      001314      012600      MOV    (SP)+,R0      ; ...

```


VAC5 CREATED BY MACRO ON 29-JUN-85 AT 02:10 PAGE 1 L 5
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
BN.LBL	000204 R	6-140 #8-188
BN.OPN	000130 R	6-138 #7-156
DSKBUF	000000 R	#5-96 8-188
FL.CHA	= ***** GX	6-136
FM.6	= 000332 R	5-72 5-73 5-74 5-75 5-76 #5-81
FM.6	= 000000	#5-71 #5-72 #5-73 #5-74 #5-75 #5-76
F2.FIL	= ***** GX	6-114 6-143
I\$BAS	= *****	8-205 8-211
L\$BEXT	= ***** GX	8-202
L\$BFLG	= ***** GX	8-194
L\$BLDZ	= ***** GX	8-196
L\$BMXZ	= ***** GX	8-197
L\$BSA	= ***** GX	8-193
MABYT	= 000354 R	#5-87 6-121
R\$SEIS	= *****	8-205 8-211
R\$S1ID	= *****	8-205 8-211
R\$SBAS	= *****	5-71 5-72 5-73 5-74 5-75 5-76
TS\$NHD	= ***** GX	5-75 5-76
\$ABFI	= 000000 RG	8-194 #6-113
\$A7FS	= ***** GX	7-159
\$BFALL	= ***** GX	*6-135 *8-213
\$BFBAS	= ***** GX	*8-193
\$BFLBN	= ***** GX	*8-191 *8-192
\$BFPTR	= ***** GX	*6-134
\$BFXFR	= ***** GX	*8-206
\$CAPR	= ***** GX	*6-121
\$CLDEQ	= ***** GX	6-142 7-171
\$CLIOS	= ***** GX	7-161 8-219
\$CLOPE	= ***** GX	7-160
\$CUR.N	= ***** GX	5-73 5-74 5-75 5-76
\$ELINE	= ***** GX	5-71 5-72 5-73 5-74 5-75 5-76
\$ERR69	= 000000 R	#5-71 7-170
\$ERR71	= 000040 R	#5-72 7-172
\$ERR72	= 000070 R	#5-73 8-220
\$ERR73	= 000144 R	#5-74 8-221
\$ERR74	= 000206 R	#5-75 8-222
\$ERR75	= 000260 R	#5-76 8-223
\$FILEN	= ***** GX	7-157
\$FLAGS	= ***** GX	6-136
\$FLAG2	= ***** GX	6-114 *6-143
\$IOSB	= ***** GX	8-219
\$LBN	= ***** GX	8-191 8-192
\$PRV.N	= ***** GX	5-71
\$REP.C	= ***** GX	5-74 5-75 5-76
\$REP.P	= ***** GX	5-72
\$RLBL	= ***** GX	8-189
\$RQ5	= ***** GX	6-131
\$RQ7	= ***** GX	6-128
\$TERR	= ***** GX	5-71 5-72 5-73 5-74 5-75 5-76
\$VALUE	= ***** GX	*6-120
\$VFLV1	= ***** GX	6-123

```

267 LOCAL DATA
268
269
270
271
272
273
274
275
276
277
278 000512
279 000512 052767 000000G 000000G
280 000520 005767 000000G
281 000524 100406
282 000526 005767 000002G
283 000532 001003
284 000534
285 000542

;+
; $UMRON - ''UMR ON'' ACTION ROUTINE
;
; INPUTS:
; FE,EXT OF $FMASK SET IF 11/70
; NETLDR DATA BLOCK+2 CONTAINS UMR BLOCK ADDRESS (IF ANY)
;
; OUTPUTS:
; F1.UMR OF $FLAG1 SET
; ERROR IF 11/70 + NO UMR'S
;
; $UMRON::
; BIS #F1.UMR,$FLAG1 ; SET THE BIT
; TST .MGEXT ; 11/70 TARGET SYSTEM ?
; BMI 20$ ; IF MI, NO
; TST $NTLHB+2 ; IS THERE A UMR BLOCK ?
; BNE 20$ ; IF NE, YES
; MSG$ ZK
20$: RETURN

```

```

93      ;+
94      $LIBR - ".LIBR" ACTION ROUTINE
95
96      : INPUTS:
97      F2.FIL SET IF VALID FILE NAME HAS BEEN SEEN
98      F2.LBR SET IF LIBRARY WAS ALREADY RESIDENT
99      $LBIAS=LIBRARY DESCRIPTOR BIAS (MAPPED ONLY)
100     $LBADR=LIBRARY DESCRIPTOR ADDRESS OR 0
101     $BFPTR=BINARY BUFFER POINTER
102
103     : OUTPUTS:
104     NONE
105
106     $LIBR::
107     000000 032767 000000G 000000G BIT #F2.FIL,$FLAG2 ; WAS VALID FILE NAME SEEN?
108     000006 001443 BEQ 40$ ; IF EQ, NO
109     000010 032767 000000G 000000G BIT #F2.LBR,$FLAG2 ; WAS LIBRARY ALREADY RESIDENT?
110     000016 001035 BNE 30$ ; IF NE, YES
111     000020 005067 000000' CLR LBLK ; INDICATE NO LIBRARY BLOCK ALLOCATED
112     000024 032767 000000G 000000G BIT #F2.LBR,$FLAG2 ; WAS LIBRARY ALREADY RESIDENT?
113     000032 001017 BNE 20$ ; IF NE, YES
114     000034 005767 000000G TST $LBADR ; IS DESCRIPTOR ADDRESS KNOWN?
115     000040 001003 BNE 10$ ; IF NE, YES
116     000042 103423 CALL $ALB ; ALLOCATE LIBRARY BLOCK
117     000046 103423 BCS 40$ ; IF CS, FAILURE
118     000050 103420 10$: CALL XFER ; TRANSFER LIBRARY INTO CORE
119     000054 005767 000000' BCS 40$ ; IF CS, FAILURE
120     000056 001413 TST LBLK ; WAS LIBRARY BLOCK ALLOCATED?
121     000062 000064 BEQ 30$ ; IF EQ, NO
122     000064 000410 CALL LINK ; LINK THIS BLOCK TO THE OTHERS
123     000070 005767 000000G BR 30$
124     000072 100402 20$: TST .MGMGE ; MAPPED TARGET SYSTEM?
125     000076 000100 BMI 25$ ; IF MI, NO
126     000100 103402 25$: CALL BASE ; CHECK BASE ADDRESS
127     000104 30$: CALL MAX14 ; ALLOW ONLY 14, ETC.
128     000110 40$: BCS 40$ ; IF CS, FAILURE
129     000112 CALL USE ; INCREMENT USE COUNT
130     000116 40$: RETURN

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54

.TITLE VAC8 - VNP TEMPLATE ACTION ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - TEMPLATE SYNTAX TPARS ACTION ROUTINES, PART 8

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

A\$\$CHK= 000000	G\$\$TSS= 000000	M\$\$NET= 000000	S\$\$YSZ= 007600	\$CLDEQ= ***** GX
A\$\$CPS= 000000	G\$\$TTK= 000000	M\$\$OVR= 000000	T\$\$KMG= 000000	\$CLIDS= ***** GX
A\$\$PRI= 000000	G\$\$WRD= 000000	N\$\$ACC= 000001	T\$\$MIN= 000000	\$CUR.N= ***** GX
A\$\$TRP= 000000	I\$\$RAR= 000000	N\$\$BUF= 000001	VF.LIB= 000001	\$ELINE= ***** GX
BFADR 000000R	002 I\$\$RDN= 000000	N\$\$LDV= 000001	V\$\$CTR= 001000	\$ERRY5 000002R
C\$\$CKP= 000000	K\$\$CNT= 177546	N\$\$MCP= 000001	V.ADD 000006	\$FNR50= ***** GX
C\$\$ORE= 000400	K\$\$CSR= 177546	N\$\$MLL= 000001	V.BIAS 000004	\$GETLB= ***** GX
C\$\$RSH= 177564	K\$\$LDC= 000000	N\$\$MOV= 000010	V.FLG 000002	\$IOSB= ***** GX
D\$\$BUG= 177514	K\$\$TPS= 000074	N\$\$NCT= 000001	V.LEN 000012	\$LBADR= ***** GX
D\$\$ISK= 000000	LD\$LP = 000000	N\$\$PEM= 000001	V.LNK 000000	\$LBIAS= ***** GX
D\$\$L11= 000001	L\$\$ASG= 000000	P\$\$P45= 000000	V.PDV 000004	\$LBN = ***** GX
D\$\$YNC= 000000	L\$\$DRV= 000000	P\$\$WRD= 000000	V.SIZ 000010	\$LBXFR 000000RG
D\$\$YNM= 000000	L\$\$P11= 000001	Q\$\$OPT= 000010	X\$\$DBT= 000000	\$LLEN = ***** GX
E\$\$XPR= 000000	L\$\$11R= 000000	R\$\$DER= 000000	\$BFALL= ***** GX	\$NTLPT= ***** GX
FMT6 = ***** GX	M\$\$CRB= 000124	R\$\$K11= 000001	\$BFBAS= ***** GX	\$READX= ***** GX
FM.6 = 000000	M\$\$CRX= 000000	R\$\$SND= 000000	\$BFLBN= ***** GX	\$TERR = ***** GX
F\$\$LVL= 000001	M\$\$FCS= 000000	R\$\$11M= 000000	\$BFXFR= ***** GX	.MGMGE= ***** GX
G\$\$TPP= 000000	M\$\$MGE= 000000	S\$\$WRG= 000000		

. ABS. 000012 000 (RW,I,GBL,ABS,OVR)
 000170 001 (RW,I,LCL,REL,CON)
 DATA 000042 002 (RW,D,LCL,REL,CON)
 Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 10750 Words (42 Pages)
 Size of core pool: 14440 Words (55 Pages)
 Operating system: RSX-11M/PLUS

Elapsed time: 00:00:09.04
 SY:VAC9.V2,[132,134]VAC9/CF -SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VAC9

```

363
364
365
366
367
368
369
370
371
372
373
374
375
376
377 000322
378 000324 012701 000000G
379 000330
380 000334 103464
381
382
383
384 000336 012702 000000G
385
386
387
388
389
390 000342 116762 000000G 000000
391 000350 016703 000000G
392 000354 006203
393 000356 006203
394 000360 110362 000001
395 000364 016762 000000G 000002
396 000372 010205
397
398 000374 016701 000000G
399 000400 116102 000000G
400 000404 116103 000000G
401
402
403
404
405
406
407
408
409
410 000410 016701 000000G
411 000414 016746 000000G
412 000420 012104
413 000422 120264 000000G
414 000426 001005
415 000430 120364 000000G
416 000434 001002
417
418
419

*** - KRBAL - ALLOCATE A KRB

INPUTS:
  $SPRI = PROCESS PRIORITY
  $SVECT = THE VECTOR ADDRESS
  $SCSR = THE CSR ADDRESS
  $SLTA = SLT ADDRESS

OUTPUTS:
  R0-R4 = DESTROYED

KRBAL: SAVRG <R5> ; SAVE REG ;[EMP01]
        MOV # $KRBLEN,R1 ; GET LENGTH OF KRB
        CALL $ALL15 ; TRY TO ALLOCATE A KRB
        BCS 101$ ; IF CS, ALLOCATION FAILURE

        ; SETUP TEMPLATE KRB
        MOV # $BFR,R2 ; POINT AT BUFFER
        .IF DF R$MPL ;[EMP01]
        SUB #K.PRM,R2 ; POINT AT KRB OFFSET 0 ;[EMP01]
        .ENDC ;[EMP01]

        MOV $SPRI,K.PRI(R2) ; STORE THE PRIORITY
        MOV $SVECT,R3 ; GET THE VECTOR ADDRESS
        ASR R3 ; DIVIDED BY 4
        ASR R3
        MOV $R3,K.VCT(R2) ; STORE THE VECTOR ADDRESS/4
        MOV $SCSR,K.CSR(R2) ; STORE THE CSR ADDRESS
        MOV R2,R5 ; SAVE BUFFER ADDRESS

        MOV $SLTA,R1 ; GET THE SLT ADDRESS
        MOV L.DDM(R1),R2 ; GET THE DDM PDV INDEX
        MOV L.CTL(R1),R3 ; GET THE CONTROLLER NUMBER

        .IF DF R$MPL ;[EMP01]
        MOV $R3,K.CON(R5) ; SET UP CONTROLLER NUMBER ;[EMP01]
        ASLB K.CON(R5) ; ... ;[EMP01]
        .ENDC ;[EMP01]

        ; STORE THE KRB ADDRESS IN EVERY SLT ON THIS CONTROLLER
        MOV $SLTMA,R1 ; GET THE SLT TABLE ADDRESS
        MOV $SLTNM,-(SP) ; GET THE NUMBER OF SLT ENTRIES
        MOV (R1)+,R4 ; GET THE NEXT SLT ADDRESS
        CMPB R2,L.DDM(R4) ; IS THIS THE SAME DEVICE ?
        BNE 20$ ; IF NE, NO .. KEEP LOOKING
        CMPB R3,L.CTL(R4) ; IS THIS THE SAME CONTROLLER ?
        BNE 20$ ; IF NE, NO .. KEEP LOOKING

        .IF DF R$MPL ;[EMP01]
        BIT #DF.MXU,$DFLAG ; KRB ADJUSTED YET ? ;[EMP01]

```

VAL16 - VNP BYTE ALLOCATION ROU MACRO V05.03b Saturday 29-Jun-85 02:12 ^{L 11}
Table of contents

5-	55	LOCAL DATA
6-	65	.ALLPR - ALLOCATE FROM PROCESS SPACE
6-	66	.ALOCB - ALLOCATE FROM DSR
7-	195	.DEAPR - DEALLOCATE FROM PROCESS
7-	196	.DEACB - DEALLOCATE FROM DSR

```

Symbol table

AB.NPV= 000001      A.NJM  000010      $$$XPR= 000000      L$$P11= 000001      P.IOC  000003
AB.TYP= 000002      A.OUTP 000010      F$$LVL= 000001      L$$11R= 000000      P.LNK  000000
AF.AS1= 000010      A.PCB  000012      GS.DEL= 000001      M$$CRB= 000124      P.MAIN 000012
AF.ESQ= 000020      A.PCBL 000000      G$$TPP= 000000      M$$CRX= 000000      P.NAM  000004
AF.LCK= 000040      A.PLEN= 000016      G$$TSS= 000000      M$$FCS= 000000      P.OWN  000026
AF.MDE= 000200      A.POWE 000004      G$$TTK= 000000      M$$MGE= 000000      P.PRI  000002
AF.NOT= 000002      A.PRI  000002      G$$WRD= 000000      M$$NET= 000000      P.REL  000014
AF.OOB= 000004      A.PRM  000012      G.CNT  000004      M$$OVR= 000000      P.SIZE 000016
AF.QUE= 000100      A.PROC 000020      G.EFLG 000006      N$$ACC= 000001      P.STAT 000030
AF.XCC= 000001      A.RECE 000016      G.GRP  000002      N$$BUF= 000001      P.SUB  000010
AK.BUF= 000200      A.REL  000000      G.LGTH= 000012      N$$LDV= 000001      P.SWSZ 000022
AK.GB1= 000202      A.RES  = 000032      G.LNK  000000      N$$MCP= 000001      P.TCB  000026
AK.GGF= 000303      A.SBUF 000022      G.STAT 000003      N$$MLL= 000001      P.WAIT 000020
AK.OCB= 000201      A.SLEN 000024      IP.ABO= 000040      N$$MOV= 000010      Q$$OPT= 000010
AS.CAA= 000006      A.SMAM 000020      IP.FAK= 000020      N$$NCT= 000001      R$$DER= 000000
AS.DEL= 000010      A.STA  000015      IP.PND= 000100      N$$PEM= 000001      R$$K11= 000001
AS.DIS= 000002      A.STAT 000010      IP.UMR= 000200      O.AST  000006      R$$SND= 000000
AS.DLT= 000001      A.TCB  000004      I$$RAR= 000000      O.EFN  000010      R$$11M= 000000
AS.EXT= 000004      A.TCBL 000006      I$$RDN= 000000      O.ESB  000012      SECTMP 000002R
AS.FPA= 000001      A.TRAN 000022      I.AST  000022      O.LGTH= 000034      S$$WRG= 000000
AS.PFA= 000004      BLSIZ  000000R      I.ATTL= 000044      O.LNK  000000      S$$YSZ= 007600
AS.RCA= 000002      CM.CDS= 000004      I.FEN  000003      O.MCRL 000002      T$$KMG= 000000
AS.REA= 000005      CM.CEN= 000003      I.FCN  000012      O.PTCB 000004      T$$MIN= 000000
AS.RED= 000001      CM.ELM= 000005      I.IOSB 000014      O.STAT 000014      UA.ACC= 000001
AS.RRA= 000003      CM.EXT= 000006      I.LGTH= 000044      PC.ALF= 000004      UA.ALL= 000400
AS.WRT= 000002      CM.IND= 000002      I.LNK  000000      PC.ALM= 001000      UA.CAL= 000100
A$$CHK= 000000      CM.INF= 000001      I.LN2  000006      PC.HIH= 000001      UA.COM= 000200
A$$CPS= 000000      CM.LKT= 000007      I.PR1  000002      PC.LOW= 000002      UA.ECH= 000004
A$$PRI= 000000      CM.MSG= 000011      I.PRM  000024      PC.NRM= 000400      UA.PRO= 000002
A$$TRP= 000000      CM.RMT= 000010      I.TCB  000004      PC.XAF= 000010      UA.PUT= 000040
A.ACC  000014      CP.DSB= 000010      I.UCB  000010      PC.XIT= 000200      UA.SPE= 000020
A.ACCE 000000      CP.EXT= 000400      I.XDBF 000040      PF.ALL= 177777      UA.TRA= 002000
A.AST  000006      CP.LGO= 000004      I.XDBL 000044      PF.INS= 000040      UA.TRN= 001000
A.BYT  000004      CP.MSG= 000002      I.XIOP 000002      PF.LOG= 000100      UA.TYP= 000010
A.CALL 000024      CP.NIO= 000100      I.XLEN= 000046      PF.REQ= 000200      V$$CTR= 001000
A.CBL  000002      CP.NUL= 000001      I.XLNK 000000      PS.APR= 000007      X$$DBT= 000000
A.CONN 000012      CP.PRIV= 000020      I.XMOD 000006      PS.CHK= 010000      $BFR  = ***** GX
A.DEQU 000002      CP.RST= 000200      I.XPBF 000022      PS.CKP= 040000      $BIAS = ***** GX
A.DIS  000002      CP.SGL= 000040      I.XPBL 000026      PS.CKR= 020000      $CRAVL= ***** GX
A.DISC 000014      C$$CKP= 000000      I.XPBV 000030      PS.COM= 000200      $GETAD= ***** GX
A.DQSR 177776      C$$ORE= 000400      I.XRBF 000012      PS.DEL= 000010      $PRAVL= ***** GX
A.FLEN 000011      C$$RSH= 177564      I.XRBL 000016      PS.DRV= 000020      $PUTAD= ***** GX
A.IBUF 000014      C.PCPL 000011      I.XTCB 000004      PS.FXD= 004000      $RADDR= ***** GX
A.ILEN 000016      C.PDPL 000010      I.XTMO 000020      PS.LIO= 001000      $RBLCK= ***** GX
A.IMAP 000012      C.PNAM 000002      I.XTTB 000032      PS.NSF= 000400      $RSIZE= ***** GX
A.INPU 000006      C.PRMT 000012      I.XTTL 000036      PS.OUT= 100000      .ALLPR 000004RG
A.IOC  000003      C.PSTS 000006      K$$CNT= 177546      PS.PER= 002000      .ALOCB 000020RG
A.IOS  000026      C.PTCB 000000      K$$CSR= 177546      PS.PIC= 000100      .ALOC1 000024RG
A.KSR5 177774      D$$BUG= 177514      K$$LDC= 000000      PS.SYS= 000040      .ALOC2 000030RG
A.LGTH= 000014      D$$ISK= 000000      K$$TPS= 000074      PS$P45= 000000      .DEACB 000226RG
A.LIN  000012      D$$L11= 000001      L$$LP  = 000000      P$$WRD= 000000      .DEAC1 000232RG
A.MAS  000004      D$$YNC= 000000      L$$ASG= 000000      P.BLKS 000016      .DEAC2 000236RG
A.MPCT 000011      D$$YNM= 000000      L$$DRV= 000000      P.BUSY 000024      .DEAPR 000212RG
A.NPR  000010

. ABS. 177776 000 (RW,I,GBL,ABS,OVR)
000434 001 (RW,I,LCL,REL,CON)

```


VAL22 CREATED BY MACRO ON 29-JUN-85 AT 02:12 PAGE 1 L 13
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
F.RSIZ	= ***** GX	*6-150 *7-227
SYSFDB	= ***** GX	*6-150 *7-227
\$BFR	= ***** GX	6-151 *6-157 6-163 6-169 6-173 *6-179 6-185 *6-187 *7-225
		*7-226 7-242 7-244 7-253 *7-256 *7-262 7-264 *7-265 *7-266
		7-272 7-273 7-276 7-278 7-279 *7-282 *7-283
\$GETAD	= ***** GX	6-150 6-172 7-238 7-247 7-277 7-281
\$PUTAD	= ***** GX	6-191 7-227 7-257 7-263 7-267 7-284
\$RADDR	= ***** GX	*6-150 *7-227
\$RBLCk	= ***** GX	*6-147 6-155 6-162 6-168 *6-169 6-177 6-184 *6-186 *7-224
		*7-237 *7-246 7-252 *7-264 7-271 7-275 *7-276 *7-280
.AMEM	000000 RG	#6-147
.DMEM	000172 RG	#7-224

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54

.TITLE VBUF - VNP END OF TASK BUFFER INITIALIZATION
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - END-OF-TASK BUFFER

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/Rsx V1.0

\$ALPOL - ALLOCATE FROM EXTENDED POOL

150	000240	016767	177536	000000G	40\$:	MOV	XAVL,\$XAVL	:	UPDATE CEX COPY OF LISTHEAD
151	000246					SWTCH1		:	RESTORE DEFAULT INPUT
152	000254					SWTCHO		:	AND OUTPUT BUFFERS
153	000262	016767	177514	000000G	50\$:	MOV	XAVL,\$XAVL	:	UPDATE CEX COPY OF LISTHEAD
154	000270	012603				MOV	(SP)+,R3	:	RESTORE REGISTER
155	000272					RETURN		:	

VCEAL CREATED BY MACRO ON 29-JUN-85 AT 02:13 PAGE 3 L 16
 MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
ASL\$	#5-59
ASR\$	#5-59
CALL	7-130 7-133 7-134 7-139 7-145 7-146 7-148 8-188 8-191 8-203
	8-205 8-215 8-222 8-224 8-232 8-237 8-250 8-257 9-286 10-320
	10-325 11-349 11-362 12-401 12-416 13-450 13-458 14-495 14-498 14-501
	14-502
CALLR	8-184
CEACC\$	#5-60
GETAD	#5-59 7-133 7-145 8-188 8-203 8-222 14-495 14-502
GETRC	#5-59 7-134 7-146 8-191 8-205 8-224 14-498 14-503
GETRV	#5-59
MAP	#5-75 8-213 8-229 8-248 8-252
PCBDF\$	#5-59 5-62
PUTAD	#5-59 7-139 7-148 8-215 8-232 8-237 8-250 8-257 14-501 14-505
PUTRC	#5-59
RAND	#7-134 7-134 #7-139 7-139 #7-146 7-146 #7-148 7-148 #8-191 8-191
	#8-205 8-205 #8-215 8-215 #8-224 8-224 #8-232 8-232 #8-237 8-237
	#8-250 8-250 #8-257 8-257 #14-498 14-498 #14-501 14-501 #14-503 14-503
	#14-505 14-505
RESMAP	#5-71 8-216 8-233
RETURN	7-155 8-264 9-297 10-326 11-363 12-422 12-427 13-464 14-512
SAVMAP	#5-67 8-192 8-211 8-221 8-242
SWTCH1	#5-59 7-121 7-128 7-151 8-190 8-204 8-223 8-260 14-496 14-506
SWTCHO	#5-59 7-122 7-129 7-152 8-214 8-231 8-236 8-249 8-255 8-259
	14-497 14-507

```

235
236
237
238
239
240
241
242
243
244
245
246 000206
247 000206
248 000212 012700 000000G
249 000216 012701 000000G
250 000222 012702 000004
251 000226
252

;+
; $INT3 - "$INT3" ACTION ROUTINE
;
; INPUTS:
;     NONE
;
; OUTPUTS:
;     R0,R1,R2=DESTROYED
;     $VECT3=CURRENT COPY OF $SIZE
;-

$INT3::
    CALL    $VFDDM           ; MAKE SURE IT'S A DDM
    MOV     #F1.INT3,R0      ; BIT TO BE SET
    MOV     #F1.INT,R1       ; BIT WHICH MUST NOT BE SET
    MOV     #4,R2            ; VECTOR INDEX
    CALLR   VFVCT            ; CHECK VECTOR AVAILABILITY

```

SYMBOL CROSS REFERENCE

CREF 04.00

SYMBOL	VALUE	REFERENCES											
\$ERR43	000316 R	#5-101	14-404										
\$ERR81	000416 R	#5-105	17-511										
\$ERR85	000444 R	#5-106	16-495										
\$EVEN	= ***** GX	12-288	14-397	15-445	15-457								
\$FILE	= 000000 RG	#6-135	12-287	14-396	15-456								
\$FILEN	= ***** GX	6-139											
\$FILEX	= 00C050 RG	#6-147											
\$FLAGS	= ***** GX	12-300	16-484	17-509	18-538								
\$FLAG1	= ***** GX	14-370	14-380	18-531									
\$FLAG2	= ***** GX	6-136											
\$GETRV	= ***** GX	18-544											
\$INT	= 000116 RG	#8-189											
\$INTX	= 000064 RG	#7-168											
\$INT1	= 000140 RG	#9-208											
\$INT2	= 000162 RG	#10-227											
\$INT3	= 000206 RG	#11-246											
\$LFILE	= 000602 RG	#14-376											
\$LFLD	= ***** GX	14-383											
\$MPLD	= ***** GX	14-373											
\$MPBT	= 000534 RG	#14-365											
\$MXADD	= ***** GX	*12-308											
\$MXLEN	= ***** GX	*12-298	*15-441										
\$MXPTB	= 000232 RG	#12-269											
\$MXSIZ	= ***** GX	*12-304											
\$MXTAB	= 000776 RG	#15-432											
\$MXVCT	= ***** GX	18-536											
\$NALL	= ***** GX	*12-307	13-338										
\$NLIB	= ***** GX	13-340											
\$NVECT	= ***** GX	*18-547											
\$OFF	= ***** GX	*15-437											
\$PRV.N	= ***** GX	5-92											
\$RADDR	= ***** GX	*18-544											
\$REP.C	= ***** GX	5-94	5-98	5-99	5-101	5-103	5-105	5-106	5-107	5-108			
		19-572											
\$RQ1	= ***** GX	12-313	14-399	15-459	16-486								
\$RQ2	= ***** GX	15-451											
\$RQ3	= ***** GX	7-169											
\$RQ4	= ***** GX	12-281	14-388										
\$SECSR	= 001164 RG	#16-481											
\$SIZE	= ***** GX	*7-174	12-286	12-289	*12-290	12-291	*12-312	14-395	14-402	*14-405			
		15-437	15-455	15-462	*15-464	*16-489	18-533						
\$TERR	= ***** GX	5-92	5-94	5-95	5-96	5-98	5-99	5-101	5-103	5-105			
		5-106	5-107	5-108									
\$TMFIX	= ***** GX	6-146											
\$VALUE	= ***** GX	6-145	*6-148	7-172	7-173	*12-274	12-276	12-278	12-284	*12-295			
		*14-374	14-384	14-386	14-391	*14-406	*15-438	15-447	15-449	15-454			
		*15-465											
		*18-533											
\$VECT1	= ***** GX	8-190	9-209	10-228	11-247	12-270	15-433	16-482					
\$VFDDM	= ***** GX	14-366											
\$VFDLG	= ***** GX	12-271	15-434	16-483	*17-508								
\$VFMUX	= 001234 RG	12-273	14-368	14-378	15-436	18-529							
\$VFSPC	= ***** GX												

```

LL          SSSSSSSS  TTTT TTTT TTTT
LL          SSSSSSSS  TTTT TTTT TTTT
          SS          TT
LL          SS          TT
LL          SS          TT
LL          SS          TT
LL          SSSSSS     TT
LL          SSSSSS     TT
LL          SSSSSS     TT
LL          SS          TT
LL          SS          TT
LL          SS          TT
LL          SS          TT
LLLLLLLLLL SSSSSSSS   TT
LLLLLLLLLL SSSSSSSS   TT

```

481 001316

RETURN

482

.DSABL LSB

483

484

485

000001

.END

VAC5 CREATED BY MACRO ON 29-JUN-85 AT 02:10 PAGE 2 M 5
SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
.BASEB	= ***** GX	5-87 6-120
.MGMGE	= ***** GX	6-117 6-125 8-208

```

287
288
289
290
291
292
293
294
295
296 000544
297 000544 042767 000000G 000000G
298 000552

```

```

: +
: $UMROF - ".UMR OFF" ACTION ROUTINE
:
: INPUTS:
:   NONE
:
: OUTPUTS:
:   F1.UMR IN $FLAG1 IS CLEARED
:
: $UMROF::
:   BIC    #F1.UMR,$FLAG1 ; CLEAR THE BIT
:   RETURN

```

```

132
133
134
135
136
137
138
139
140
141
142
143 000120 005767 000000G XFER: TST ,MGMGE ; MAPPED TARGET SYSTEM?
144 000124 100414 BMI 5$ ; IF MI, NO
145 000126 032767 017777 000000G BIT #17777,$BFBAS ; BUILT TO SOME 4K BOUNDARY?
146 000134 001041 BNE 101$ ; IF NE, NO
147 000136 026767 000000G 000000G CMP $VALUE,$BFBAS ; BUILT FOR CORRECT APR?
148 000144 001412 BEQ 10$ ; IF EQ, YES
149 000146 EMSG$ Y2 ; PRINT WARNING MESSAGE
150 000154 000406 BR 10$
151 000156 005767 000000G 5$: TST $BFBAS ; BUILT FOR BASE ADDRESS OF 0?
152 000162 001403 BEQ 10$ ; IF EQ, YES
153 000164 EMSG$ Y7 ; PRINT WARNING MESSAGE
154 000172 103417 10$: CALL MAX14 ; ALLOW ONLY 14 ALLOCS/LIBRS
155 000176 016701 000000G BCS 20$ ; IF CS, TOO MANY ALREADY
156 000200 032767 000000G 000000G MOV $BFBALL,R1 ; GET ALLOCATION SIZE NEEDED
157 000204 001003 BIT #F1.UMR,$FLAG1 ; IS UMR MAPPING NEEDED?
158 000212 000402 BNE 15$ ; IF NE, YES
159 000214 CALL $ALN18 ; ALLOCATE FROM NON-MAPPED AREA
160 000220 BR 18$
161 000222 15$: CALL $ALL18 ; ALLOCATE FROM NT POOL
162 000226 103411 18$: BCS 111$ ; IF CS, FAILURE
163 000230 CALL $LBXFR ; TRANSFER LIBRARY INTO CORE
164 000234 103413 BCS 121$ ; IF CS, FAILURE
165 000236 20$: RETURN
166
167
168
169
170 000240 101$: CALL $LBERR ; CLEANUP AFTER AN ERROR
171 000244 EMSG$R Y3 ; LIBRARY BASE ADDRESS NOT MULTIPLE OF 4K
172 000252 111$: CALL $LBERR ; CLEANUP AFTER AN ERROR
173 000256 EMSG$R Y4 ; LIBRARY ALLOCATION FAILURE
174 000264 016701 000000G 121$: MOV $BFBALL,R1 ; GET LIBRARY SIZE
175 000270 CALL $DEA18 ; DE-ALLOCATE LIBRARY
176 000274 CALL $LBERR ; CLEANUP AFTER AN ERROR
177 000300 000261 SEC ; RETURN C-BIT SET
178 000302 RETURN

```

```

56                                     :***
57                                     : LIBRARY MACROS
58                                     :***
59                                     .MCALL  EMSG$R,NTLR$
60                                     .ENABL  LC
61
62                                     :***
63                                     : LOCAL DATA
64                                     :***
65
66 000000                             .PSECT  DATA,D
67
68                                     :
69                                     : ERROR MESSAGE (FORMAT STRING ADDRESS DEFINED GLOBALLY IN 'TMPAC7')
70                                     :
71 000000                             NTLR$ ,Y1,2,$TERR,$REP.C,$ELINE,<Library block allocation failure.>
72
73                                     .EVEN
74
75 000000                             .PSECT

```

VAC9 CREATED BY MACRO ON 29-JUN-85 AT 02:11 PAGE 1 M 9

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
BFADR	= 000000 R	#5-92 *6-115 6-172 6-189
FM16	= ***** GX	5-97
FM.6	= 000000	#5-97
IS\$AS	= *****	6-179
RS\$EIS	= *****	6-179
RS\$MPL	= *****	5-71 6-123 7-205
RS\$11D	= *****	6-179
SS\$BAS	= *****	5-97
\$BFALL	= ***** GX	6-173
\$BFBAS	= ***** GX	6-177
\$BFLBN	= ***** GX	6-116 6-117
\$BFXFR	= ***** GX	6-118
\$CLDEQ	= ***** GX	6-121
\$CLIOS	= ***** GX	6-187
\$CUR.N	= ***** GX	5-97
\$ELINE	= ***** GX	5-97
\$ERRY5	= 000002 R	#5-97 6-188
\$FNR50	= ***** GX	6-169 6-170 6-171
\$GETLB	= ***** GX	6-164
\$IOSB	= ***** GX	6-187
\$LBADR	= ***** GX	6-168
\$LBIAS	= ***** GX	6-163
\$LBN	= ***** GX	*6-116 *6-117
\$LBXFR	= 000000 RG	#6-114
\$LEN	= ***** GX	*6-118
\$NTLPT	= ***** GX	6-165
\$READX	= ***** GX	6-119
\$TERR	= ***** GX	5-97
.MGMGE	= ***** GX	6-175

```

420                                     BNE      15$      ; IF YES - BRANCH      ;[MP01]
421                                     SUB      #K.PRM,R0      ; ADJUST ADDRESS TO OFFSET 0      ;[MP01]
422                                     BIS      #DF.MXU,$DFLAG      ; ONLY ADJUST ONCE      ;[MP01]
423                                     15$:                                     ;
424                                     .ENDC                                     ;[MP01]
425
426 000436 010064 000000G      20$: MOV      R0,L.KRBA(R4)      ; STORE THE KRB ADDRESS IN THE SLT
427 000442 005316      DEC      (SP)      ; LOOP <SLTNM> TIMES
428 000444 003365      BGT      10$
429 000446 005726      TST      (SP)+      ; CLEAN UP STACK
430
431                                     .IF DF $$$MPL      ;[MP01]
432                                     BIC      #DF.MXU,$DFLAG      ; RESET MUX KRB ADJUSTMENT FLAG      ;[MP01]
433                                     CLRB     K.IOC(R5)      ; INITIALIZE I/O COUNT      ;[MP01]
434                                     MOV      #KS.OFL,K.STS(R5)      ; SET OFFLINE STATUS      ;[MP01]
435                                     CLRB     K.HPU(R5)      ; CLEAR HIGHEST UNIT NUMBER      ;[MP01]
436                                     CLR      K.OWN(R5)      ; NO OWNING CONTROLLER      ;[MP01]
437
438                                     MOV      $URM,K.URM(R5)      ; STORE THE UNIBUS RUN MASK      ;[MP01]
439                                     CLR      K.CRQ(R5)      ; INITIALIZE CONTROLLER REQUEST QUEUE      ;[MP01]
440                                     MOV      R0,R1      ; ...
441                                     ADD      #K.CRQ,R1      ; ...
442                                     MOV      R1,K.CRQ+2(R5)      ; ...
443                                     CLR      R1      ; INIT REG      ;[MP01]
444
445                                     MOV      K.CON(R5),R1      ; GET CONTROLLER NUMBER      ;[MP01]
446                                     ADD      $CTBAD,R1      ; CALCULATE START OF CTB      ;[MP01]
447                                     MOV      R0,L.KRB(R1)      ; STORE KRB ADDRESS IN CTB      ;[MP01]
448                                     SUB      #-<K.PRM>,R0      ; SET ADDRESS TO START OK KRB      ;[MP01]
449                                     .ENDC
450
451 000450      SWITCHI      ;[MP01]
452 000456      SWTCHO      ;[MP01]
453 000464      PUTRC      R0,#$KRBLN      ; WRITE KRB TO IMAGE      ;[MP01]
454 000502      RESRG      <R5>      ; RESTORE REG      ;[MP01]
455
456 000504      RETURN
457
458      ; ERROR CON ITIONS
459
460 000506      101$:  MSG$R  9C      ; KRB ALLOCATION FAILURE
461
462

```

M 11

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52

.TITLE VAL16 - VNP BYTE ALLOCATION ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - SINGLEWORD ADDRESSABLE ALLOCATION ROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M v4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

N 11

VAL16 - VNP BYTE ALLOCATION ROU MACRO V05.03b Saturday 29-Jun-85 02:12 ^{M 12} Page 9-3
Symbol table

Errors detected: 0

*** Assembler statistics

work file reads: 0
work file writes: 0
Size of work file: 16005 Words (63 Pages)
Size of core pool: 17608 Words (67 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:17.14
SY: VAL16.V2,[132,134]VAL16/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VAL16

VAL22 CREATED BY MACRO ON 29-JUN-85 AT 02:12 PAGE 2 M 13
 MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
ASL\$	#6-144
CALL	6-150 6-172 6-191 7-227 7-238 7-247 7-257 7-263 7-267 7-277
	7-281 7-284
ERBLK\$	#6-144
ERMSG\$	#6-144
ERRPT\$	#6-144
GETAD	#6-144 6-150 6-172 7-238 7-247 7-277 7-281
PUTAD	#6-144 6-191 7-227 7-257 7-263 7-267 7-284
RAND	#6-150 6-150 #6-172 6-172 #6-191 6-191 #7-227 7-227 #7-238 7-238
	#7-247 7-247 #7-257 7-257 #7-263 7-263 #7-267 7-267 #7-277 7-277
	#7-281 7-281 #7-284 7-284
RETURN	6-194 7-285

```

56      ;***
57      ; THIS MODULE CONTAINS A BUFFER POOL WHICH IS LOCATED IN THE PSECT
58      ; "POOL". THIS PSECT ASSIGNMENT PLACES THE POOL AT THE END OF THE
59      ; TASK IMAGE. THUS, ALL SPACE BETWEEN THE START OF THE PSECT AND THE
60      ; END OF THE TASK'S PARTITION CAN BE ASSIGNED TO THE POOL. ON A MAPPED
61      ; SYSTEM, THE POOL CAN BE INCREASED BY INSTALLING THE TASK WITH THE
62      ; "/INC=" SWITCH. ANY ONE-TIME INITIALIZATION CODE CAN BE ASSIGNED TO
63      ; THE POOL BY NAMEING ITS PSECT "****", WHERE "****" MUST BE HIGHER IN
64      ; COLLATING SEQUENCE THAN "POOL". THE RSX11M ALLOCATION ROUTINES "$ALOC1"
65      ; AND "$DEACT" FROM THE MODULE "CORAL" MAY BE USED IN CONJUNCTION WITH
66      ; THIS POOL.
67
68      ; IN AN OVERLAID ENVIRONMENT, THE MODULE VBUF MUST APPEAR IN THE ODL
69      ; AS A CO-OVERLAY AND NOT AS AN ORDINARY OVERLAY. E.G.:
70
71      .ROOT    ROOT=*(OVR1,VBUF)
72
73      ;***
74
75      .MCALL   GREG$$,GTSK$$
76
77      .PSECT   DATA,D
78      000000
79
80      ;
81      ; POOL LIMITS
82
83      000000 000000      $XXBUF:::WORD 0          ; STARTING ADDRESS
84      000002 000000      $XXLEN:::WORD 0         ; LENGTH IN BYTES
85
86      ;
87      ; "CORAL" POINTERS
88
89      000004 000003      .WORD 3          ; ROUNDING FACTOR
90      000006 000000      $XXAVL:::WORD 0       ; POINTER TO FIRST FREE BLOCK
91      00C010 000000      .WORD 0

```

157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213

000274
000274 032700 000001
000300 001002
000302
000306 010446
000310 010546
000312 010005
000314
000324 012767 000004 000000G
000332
000340
000350
000354 012703 000000'
000360 061301
000362 042301
000364 001002
000366 000167 000406
000372 016713 000000G
000376 005746
000400 011367 177412
000404 016716 177406
000410
000422
000430
000440 016703 177352
000444 001402
000446 020503
000450 103355
000452
000456 010367 177324
000462

```

.SBTTL $DEPOL - DEALLOCATE NETWORK POOL
*** - $DEPOL - DEALLOCATE STORAGE BACK TO THE NETWORK POOL

THIS ROUTINE IS CALLED TO DEALLOCATE A CORE BUFFER IN EXTENDED POOL.
THE BLOCK IS INSERTED INTO THE FREE BLOCK CHAIN BY CORE ADDRESS. IF
AN ADJACENT BLOCK IS CURRENTLY FREE, THEN THE TWO BLOCKS ARE MERGED
AND INSERTED IN THE FREE BLOCK CHAIN.

INPUTS:

R0=UNMAPPED ADDRESS OF THE CORE BUFFER TO BE DEALLOCATED.
R1=SIZE OF THE CORE BUFFER TO DEALLOCATE IN BYTES.

OUTPUTS:

THE CORE BLOCK IS MERGED INTO THE FREE CORE CHAIN BY CORE
ADDRESS AND IS MERGED IF NECESSARY WITH ADJACENT BLOCKS.

REGISTERS:

R2, R3 ARE MODIFIED

$DEPOL:
BIT #1,R0      : IS THE BLOCK IN EXTENDED POOL ?
BNE 10$       : IF NE, YES
CALLR $DEA16   : ELSE, DEALLOCATE RSX CORE BLOCK
10$: MOV R4,-(SP) : SAVE SOME REGISTERS
      MOV R5,-(SP)
      MOV R0,R5
      CEACCS R0,R4 : COPY UNMAPPED ADDRESS
      MOV #4,$RSIZE : CONVERT TO MAPPED ADDRESS
      SWITCHJ #LOCBUF : READ IN FOUR BYTES
      GETAD R4      : SET UP LOCBUF AS INPUT BUFFER
      SAVMAP       : READ CURRENT BLOCK (ONLY 4 BYTES)
      MOV #XAVL-2,R3 : SAVE CURRENT MAPPING
      ADD (R3),R1    : POINT TO ALLOCATION MASK WORD
      BIC (R3)+,R1   : ROUND TO NEXT BOUNDARY
      BNE 15$       : CLEAR EXCESS
      JMP 70$       : IF SOMETHING TO RETURN - BRANCH
15$: MOV $XAVL,(R3) : IF EQ, NOTHING TO RETURN
      TST -(SP)     : COPY CEX ALLOCATION LISTHEAD
      MOV (R3),LOCBF2 : RESERVE A WORD ON THE STACK
                        : SAVE FIRST ENTRY IN LISTHEAD

20$: MOV LOCBF2,(SP) : SAVE UNMAPPED ADDRESS IN RESERVED WORD
      CEACCS LOCBF2,R2 : CONVERT TO MAPPED ADDRESS
      SWITCHJ #LOCBF2 : SET UP LOCBF2 AS INPUT BUFFER
      GETAD R2        : READ LISTHEAD INTO LOCBF2
      MOV LOCBF2,R3   : GET ADDRESS OF NEXT BLOCK
      BEQ 40$         : IF EQ END OF CHAIN
      CMP R5,R3       : BLOCK GO BEFORE HERE?
      BHS 20$        : IF HIS NO

30$: MOV SAVMAP      : SAVE CURRENT MAPPING
      MOV R3,LOCBUF  : ASSUME BLOCKS NOT ADJACENT
      MAP 4(SP)      : MAP TO BLOCK BEING DEALLOCATED

```

```

      SSSSSSSS   TTTTTTTTTT
      SSSSSSSS   TTTTTTTTTT
          SS           TT
          SS           TT
          SS           TT
          SS           TT
              SSSSSS   TT
              SSSSSS   TT
                      SS   TT
                      SS   TT
                      SS   TT
                      SS   TT
          SSSSSSSS   TT
          SSSSSSSS   TT
          SSSSSSSS   TT
          SSSSSSSS   TT

```

```

254      ;+
255      $MXPTB - ".MXPTB" ACTION ROUTINE
256
257      : INPUTS:
258      $SMUX/$VALUE = FILE REPEAT COUNT
259      $FILE=SECONDARY FILE NAME
260
261      : OUTPUTS:
262      $MXADD=MUX TABLE ADDRESS
263      $MXLEN=SIZE OF SECONDARY TEMPLATE FILE
264      $MXSIZE=SIZE OF ALL MUX TABLES
265      R0,R1,R2=DESTROYED
266
267      :-
268
269      $MXPTB::
270      CALL    $VFDDM      ; MAKE SURE IT'S A DDM
271      CALL    $VFMUX      ; AND A MUX LINE
272      MOV     #F1.MXT,R0  ; MAKE SURE THIS IS THE ONLY OCCURANCE
273      CALL    $.PC        ; ...
274      MOV     $$JX,$VALUE ; DO IT <UNIT COUNT> TIMES
275
276      CMP     #256,$VALUE ; IS COUNT > 256. ?
277      BLO     101$        ; IF LO, YES
278      TST     $VALUE      ; IS COUNT 0 ?
279      BEQ     30$         ; IF EQ, YES
280      CALL    MAX14       ; ALLOW ONLY 14. ALLOCATIONS
281      CALL    $RQ4        ; REQUEST BINARY FILE
282      BCS     30$         ; IF CS, NO ACCESS
283      MOVB    #.MXPTB,(R2)+ ; STORE FUNCTION CODE
284      MOVB    $VALUE,(R2)+ ; AND FILE REPEAT COUNT
285      MOV     R2,-(SP)     ; SAVE POINT TO BINARY INFO.
286      MOV     $SIZE,-(SP) ; SAVE CURRENT LINE TABLE SIZE
287      CALL    $FILE       ; PROCESS SECONDARY TEMPLATE
288      CALL    $EVEN       ; AUTOMATIC WORD RE-ALIGNMENT
289      MOV     $SIZE,R0     ; SAVE CURRENT POSITION IN BINARY BUFFER
290      MOV     (SP)+,$SIZE  ; RESTORE SIZE FROM START OF FILE
291      SUB     $SIZE,R0     ; CALCULATE SIZE OF FILE
292      MOV     (SP)+,R2     ; RESTORE POINTER TO BINARY BUFFER
293      CLR     R1           ; INITIALIZE DATA BASE SIZE
294      ADD     R0,R1        ; CALCULATE DATA BASE SIZE
295      DEC     $VALUE       ; BY <FILE COUNT> * <FILE SIZE>
296      BNE     10$         ; ...
297      NEG     R0           ; GET FILE SIZE AS NEGATIVE
298      MOV     R0,$MXLEN    ; SAVE MUX TABLE LENGTH
299      CLR     R0           ; ASSUME MUX UPDATE
300      BIT     #FL.CHA,$FLAGS ; MUX UPDATE ?
301      BNE     20$         ; IF NE, YES
302      INC     R1           ; ROUND UP TO MULTIPLE OF 2
303      BIC     R1,R1       ; ...
304      MOV     R1,$MXSIZE  ; SAVE MUX TABLE SIZE
305      CALL    $ALL16      ; TRY TO ALLOCATE MUX TABLES
306      BCS     111$        ; IF CS, ALLOCATION ERROR
307      INCB    $NALL       ; TALLY SUCCESSFUL ALLOCATION
308      MOV     R0,$MXADD   ; SAVE ADDRESS OF 'TMPVCT'
309      MOVB    R0,(R2)+    ; STORE ADDRESS
310      SWAB    R0          ; ...

```

SYMBOL CROSS REFERENCE

CREF 04.00

SYMBOL	VALUE	REFERENCES
\$\$MTP	= GX	14-374
\$\$MUX	= GX	12-274 15-438
\$\$VECT	= GX	18-535
.EOF	= GX	12-315 14-401 15-461
.FILE	= GX	15-446
.INT	= GX	7-171
.LFILE	= GX	14-382
.MPTAB	= GX	14-372
.MXPTB	= GX	12-283
.MXTAB	= GX	15-439
.NS0	= GX	5-80
.NS1	= GX	5-81
.NS2	= GX	5-82
.NS3	= GX	5-83
.NS4	= GX	5-84
.NS5	= GX	5-85
.NS6	= GX	5-86
.NS7	= GX	5-87
.SECSR	= GX	16-488

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54

.TITLE VAC4 - VNP TEMPLATE ACTION ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - TEMPLATE SYNTAX TPARS ACTION ROUTINES, PART 4

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

```

Symbol table
ALLERR 000712R      G$SWRD= 000000      N$SMCP= 000001      $BFALL= ***** GX      $FLAG1= ***** GX
A$SCHK= 000000      I$SRAR= 000000      N$SMML= 000001      $BFBAS= ***** GX      $FLAG2= ***** GX
A$SCPS= 000000      I$SRDN= 000000      N$SMOV= 000010      $BFLBN= ***** GX      $IOSB = ***** GX
A$SPRI= 000000      K$ISAR= ***** GX      N$SMCT= 000001      $BFPTR= ***** GX      $LBN  = ***** GX
A$STRP= 000000      K$CNT= 177546      N$SPEM= 000001      $BFR  = ***** GX      $LLEN = ***** GX
C$SCKP= 000000      K$CSR= 177546      P$P45= 000000      $BFXFR= ***** GX      $NALL = ***** GX
C$SORE= 000400      K$LDC= 000000      Q$SPT= 000010      $BINF= 000004RG      $NLIB = ***** GX
C$SRSH= 177564      K$TPS= 000074      REP6  001100R      $BING= 000000RG      $PUTAD= ***** GX
D$SBUG= 177514      LD$LP= 000000      R$SK11= 000001      $CAPR= ***** GX      $RADDR= ***** GX
D$ISK= 000000      L$ASG= 000000      R$SND= 000000      $CLDEQ= ***** GX      $RBLCK= ***** GX
D$SL1= 000001      L$DRV= 000000      R$SK11= 000001      $CLIOS= ***** GX      $READX= ***** GX
D$SYNC= 000000      L$P11= 000001      R$SND= 000000      $CORE  000510RG      $REP.C= ***** GX
D$SYNM= 000000      L$11R= 000000      R$S11M= 000000      $CORE1 000312RG      $RQ5  = ***** GX
E$XPR= 000000      MAPBYT 000023R      R$S11M= 000000      $CORE2 000400RG      $RQ7  = ***** GX
FLCHA= ***** GX      MAPP  000002R      S$WGR= 000000      $CORK  000504RG      $RSIZE= ***** GX
FMT2  000360R      MAX14  001046R      S$YSZ= 007600      $CUR.N= ***** GX      $SCOM 000720RG
FMT6  000377R      M$CRB= 000124      T$KMG= 000000      $CVAL = ***** GX      $SIZE = ***** GX
FM.2  = 000000      M$CRX= 000000      T$MIN= 000000      $ELINE= ***** GX      $TERR = ***** GX
FM.6  = 000000      M$FCS= 000000      U$SAR= ***** GX      $ERRZA 000144R      $TWARN= ***** GX
F$SLVL= 000001      M$MGE= 000000      V$CTR= 001000      $ERR26 000034R      $VALUE= ***** GX
F1.UMR= ***** GX      M$NET= 000000      X$DBT= 000000      $ERR39 000102R      $VFLV1= ***** GX
F2.FIL= ***** GX      M$OVR= 000000      ZER16  001200R      $ERR76 000200R      $BASEB= ***** GX
G$STPP= 000000      N$ACC= 000001      ZER18  001136R      $ERR77 000240R      $CORE = ***** GX
G$TSS= 000000      N$BUF= 000001      $ALL16= ***** GX      $ERR78 000300R      $MGMGE= ***** GX
G$TTK= 000000      N$LDV= 000001      $ALN18= ***** GX      $FLAGS= ***** GX      $SCOM = ***** GX

```

```

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
        001320 001 (RW,I,LCL,REL,CON)
DATA 000420 002 (RW,C,LCL,REL,CON)
Errors detected: 0

```

*** Assembler statistics

```

Work file reads: 0
Work file writes: 0
Size of work file: 10552 Words ( 42 Pages)
Size of core pool: 14440 Words ( 55 Pages)
Operating system: RSX-11M/PLUS

```

```

Elapsed time: 00:00:16.39
SY:VAC4.V2,[132,134]VAC4/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VAC4

```


VACS CREATED BY MACRO ON 29-JUN-85 AT 02:10 PAGE 3 N 5

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

ASL\$	#5-59	8-205	8-211						
CALL	6-123	6-128	6-131	6-138	6-140	6-142	7-159	7-160	7-171 8-189
EMSG\$	#5-59								
EMSG\$R	#5-59	7-170	7-172	8-220	8-221	8-222	8-223		
NTLR\$	#5-59	5-71	5-72	5-73	5-74	5-75	5-76		
RETURN	6-144	7-165	8-214						

```

300                                     .SBTTL $X2CHW - .X2CHW ACTION ROUTINE
301                                     .SBTTL $X3CHW - .X3CHW ACTION ROUTINE
302
303                                     ;+
304                                     ;
305                                     ;$X2CHW - .X2CHW ACTION ROUTINE
306                                     ;$X3CHW - .X3CHW ACTION ROUTINE
307
308                                     ; INPUTS:
309                                     ; $VALUE - VALUE OF FIRST OPERAND (INDICATES WHICH PARAMETER
310                                     ; FROM X2P$DF OR X3P$DF TO USE)
311
312                                     ; OUTPUTS:
313                                     ; R1 DESTROYED
314
315                                     ; -
316
317 000554 012701 000000G $X2CHW: .ENABL LSB ; GET FUNCTION CODE
318 000560 026727 000000G 000002 .MOV #.X2CH1,R1 ; LEVEL 2 BLOCK SIZE?
319 000566 001410 .CMP $VALUE,#X2BLK ; YES, SPECIAL CASE (NOT IN RANGE)
320 000570 026727 000000G 000004 .BEQ 5$ ; IS VALUE WITHIN RANGE?
321 000576 103421 .CMP $VALUE,#X2WLO ; BR IF NO
322 000600 026727 000000G 000005 .BLO 10$ ; WITHIN RANGE?
323 000606 101015 .CMP $VALUE,#X2WHI ; BR IF NO
324 000610 5$: .BHI 10$ ; MAKE SURE IT'S NOT AN LLC
325 000614 .CALL $VXLLC
326 .BR 30$
327 000616 012701 000000G $X3CHW: .MOV #.X3C01,R1 ; GET FUNCTION CODE
328 000622 026727 000000G 000001 .CMP $VALUE,#X3WLO ; IS OPERAND WITHIN RANGE?
329 000630 103404 .BLO 10$ ; BR IF NO
330 000632 026727 000000G 000002 .CMP $VALUE,#X3WHI ; WITHIN RANGE?
331 000640 101403 .BLOS 20$ ; BR IF YES
332 000642 .EMSG$ 91 ; INVALID OPERAND
333 000650 20$: .CALL $VFLLC ; MAKE SURE IT IS AN LLC
334 000654 066701 000000G 30$: .ADD $VALUE,R1 ; POINT TO CORRECT FUNCTION CODE
335 000660 005301 .DEC R1 ; OPERAND STARTS AT 1
336 000662 .CALL $R01 ; REQUEST BINARY BUFFER
337 000666 110122 .MOVB R1,(R2)+ ; STORE FUNCTION CODE
338 000670 062767 000002 000000G .ADD #2,$SIZE ; ADJUST DATA BASE SIZE
339 000676 .RETURN
340 .DSABL LSB

```

180
181
182
183
184
185
186
187
188
189

190 000304 116700 000000G
191 000310 116701 000000G
192 000314 060001
193 000316 022701 000015
194 000322 103401
195 000324

196
197
198
199

200 000326
201 000332

MAX14 - ALLOW ONLY 14. ALLOCATIONS/LIBRARIES

INPUTS:
\$NALL=NUMBER OF ALLOCATIONS
\$NLIB=NUMBER OF LIBRARIES

OUTPUTS:
C-BIT=SUCCESS/FAILURE

MAX14: MOVB \$NALL,R0 ; GET NUMBER OF ALLOCATIONS
MOVB \$NLIB,R1 ; AND NUMBER OF LIBRARIES
ADD R0,R1 ; ADD THEM TOGETHER
CMP #14,R1 ; ARE WE AT THE MAXIMUM ALREADY?
BCS 101\$; IF CS, YES
RETURN

ERROR CONDITION

101\$: CALL \$LBERR ; CLEANUP AFTER AN ERROR
MSG\$R Y6 ; ALREADY 14. ALLOCS/LIBRS

```

77
78
79
80
81
82
83
84
85
86
87 000000
88 000000 012701 000001
89 000004 005767 000000G
90 000010 100002
91 000012 012701 000100
92 000016
93 000022 103431
94 000024 010067 000000G
95 000030 010067 000000G
96 000034 012767 000004 000000G
97 000042 012701 000000G
98 000046 005021
99 000050 005021
100 000052 005061 000014
101 000056 005061 000030
102 000062 005061 000044
103 000066 005061 000060
104 000072 010067 000000G
105 000076
106 000102 000241
107 000104
108
109
110
111
112 000106
113 000112
114
115 000001

;+
; $ALB - ALLOCATE LIBRARY BLOCK
;
; INPUTS:
;     NONE
;
; OUTPUTS:
;     C-BIT=SUCCESS/FAILURE
;     $LBIAS, $LBADR=DESCRIPTOR ADDRESS
;
$ALB::
MOV     #1,R1           ; ASSUME MAPPED TARGET
TST     .MGMGE          ; MAPPED TARGET SYSTEM?
BPL     10$            ; IF PL, YES
MOV     #100,R1         ; UNMAPPED ALLOCATION SIZE
CALL    $ALN18
BCS     101$           ; IF CS, FAILURE
MOV     R0,LBLK         ; SAVE LIBRARY BLOCK ADDRESS
MOV     R0,$LBIAS       ; SET DESCRIPTOR BIAS
MOV     #4,$LBADR       ; AND OFFSET TO FIRST DESCRIPTOR
MOV     #.LBUFF,R1      ; ADDRESS $GETLB BUFFER AREA
CLR     (R1)+            ; NO NEXT BLOCK IN LIST
CLR     (R1)+            ; NO DESCRIPTORS IN BLOCK YET
CLR     12.(R1)         ; INDICATE NO OTHER LIBRARIES
CLR     24.(R1)
CLR     36.(R1)
CLR     48.(R1)
MOV     R0,.LBIAS       ; SET BIAS FOR $GETLB
CALL    $PUTLB          ; WRITE JUST-INITIALIZED BLOCK
CLC                     ; RETURN WITH C-BIT CLEAR
RETURN

;
; ERROR CONDITION
;
101$:  CALL    $LBERR      ; CLEANUP AFTER AN ERROR
      EMMSG$R Y1         ; LIBRARY BLOCK ALLOCATION FAILURE

      .FND

```

VAC9 CREATED BY MACRO ON 29-JUN-85 AT 02:11 PAGE 2 N 9
MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

ASR\$	#5-58	6-179		
CALL	6-119	6-121	6-164	6-188
CEACC\$	#5-58			
EMSG\$	#5-58	6-188		
NTLER\$	#5-58	5-97		
PUTAD	#5-58			
RESRG	#5-58			
RETURN	6-182	6-190		
SAVRG	#5-58			
VECDF\$	#5-58	5-61		

```

464      ;+
465      ;STMCHK - CHECK FOR CORRECT DDM INTERRUPT LINKAGE
466
467      INPUTS:
468          $NVECT=NUMBER OF ".INT" OPERATORS PROCESSED
469
470      OUTPUTS:
471          R0,R1=DESTROYED
472          F1.ERR SET IF INTERRUPT LINKAGE IS INCORRECT
473      :-
474
475      000514 016700 000000G      STMCHK::MOV      $FLAG1,R0      ; GET TEMPLATE FILE FLAGS
476      000520 109443      BMI      30$      ; IF M1, ERRORS - DON'T CHECK
477      000522 032767      C00000G 000000G      BIT      #F1.DDM,$FLAGS      ; ARE WE LOADING A DDM?
478      000530 001423      BEQ      10$      ; IF EQ, NO - DON'T CHECK
479      000532 016701 000000G      MOV      $PDVA,R1      ; GET PDV ADDRESS
480      000536 032761 000000G 000000G      BIT      #ZF.PSE,Z.FLG(R1) ; IS IT A PSEUDO-DEVICE?
481      000544 001015      BNE      10$      ; BR IF YES
482      000546 016701 000000G      MOV      $NVECT,R1      ; GET NUMBER OF ".INT" OPERATORS
483      000552 001427      BEQ      101$      ; IF EQ, NONE SPECIFIED
484      000554 032700 000000G      BIT      #F1.INT,R0      ; WAS THE ".INT" OPERATOR SEEN?
485      000560 001007      BNE      10$      ; IF NE, YES - KNOWN TO BE CORRECT
486      000562 006301      ASL      R1      ; MAKE A WORD INDEX
487      000564 010046      MOV      R0,-(SP)      ; EXTRACT ".INT" BITS FROM $FLAG1
488      000566 042716 000000G      BIC      #^C<F1.IN1!F1.IN2!F1.IN3>,(SP) ;
489      000572 026126 000000G      CMP      ITAB-2(R1),(SP)+ ; RIGHT COMBINATION OF Bits SET?
490      000576 001020      BNE      111$      ; IF NE, NO
491      000600 032700 000000G      10$: BIT      #F1.MPL,R0      ; ".MPLHD" OPERATOR SEEN?
492      000604 001403      BEQ      20$      ; IF EQ, NO
493      000606 032700 000000G      BIT      #F1.MPT,R0      ; ".MPTAB" OPERATOR SEEN?
494      000612 001415      BEQ      20$      ; IF EQ, NO
495      000614 032700 000000G      20$: BIT      #F1.LFL,R0      ; ".LFLHD" OPERATOR SEEN?
496      000620 001403      BEQ      30$      ; IF EQ, NO
497      000622 032700 000000G      BIT      #F1.LFI,R0      ; ".LFILE" OPERATOR SEEN?
498      000626 001412      BEQ      131$      ; IF EQ, NO
499      000630      30$: RETURN
500
501      ;
502      ; ERROR CONDITIONS
503
504      000632      101$: MSG$R 35      ; NO INTERRUPT LINKAGE
505      000640      111$: MSG$R 36      ; INCOMPLETE INTERRUPT LINKAGE
506      000646      121$: MSG$R 37      ; ".MPTAB" OPERATOR MISSING
507      000654      131$: MSG$R 84      ; ".LFILE" OPERATOR MISSING
508

```

54
55
56
57
58
59
60
61
62
63

.SBTTL LOCAL DATA

: LOCAL DATA

000000
000002

BLSIZ: .BLKW 1
SECTMP: .BLKW 1

; BLOCK SIZE OF PREVIOUS BLOCK
; I/O BUFFER FOR FREE SECONDARY POOL

VAL16 CREATED BY MACRO ON 29-JUN-85 AT 02:12 PAGE 1 N 12
 SYMBOL CROSS REFERENCE CREF 04.00

SOURCE SYMBOL	VALUE	REFERENCES
BLS1Z	000000 R	#5-61 *6-130 6-156
R\$MPL	= *****	6-134 6-158 6-173
		7-337 8-345 9-413
SECTMP	000002 R	#5-62
\$BFR	= ***** GX	*6-127 6-130 6-131 6-145 *6-147 6-151 *6-156 6-186 7-232
		*7-321
\$BIAS	= ***** GX	6-109 7-220
\$CRAVL	= ***** GX	6-112 7-229
\$GETAD	= ***** GX	6-135 7-243 7-259 7-309
\$PRAVL	= ***** GX	6-108 7-219
\$PUTAD	= ***** GX	6-159 6-175 7-290 7-324
\$RADDR	= ***** GX	*6-135 *6-159 *6-175 *7-243 *7-259 *7-290 *7-309 *7-324
\$RBLCK	= ***** GX	*6-109 *6-114 6-116 *7-220 *7-231
\$RSIZE	= ***** GX	*6-117 *7-235
.ALLPR	000004 RG	#6-108
.ALOCB	000020 RG	#6-112
.ALOC1	000024 RG	#6-114
.ALOC2	000030 RG	6-110 #6-115
.DEACB	000236 RG	#7-229
.DEAC1	000232 RG	#7-231
.DEAC2	000236 RG	7-227 #7-232
.DEAPR	000212 RG	#7-219

★ ★ FILE ★ ★ ID ★ ★ VB10

```

VV      VV  88888888      111111      000000
VV      VV  88888888      111111      000000
VV      VV  88      88      11      00      00
VV      VV  88      88      11      00      00
VV      VV  88      88      11      00      00
VV      VV  88      88      11      00      00
VV      VV  88888888      11      00      00
VV      VV  88888888      11      00      00
VV      VV  88      88      11      00      00
VV      VV  88      88      11      00      00
VV      VV  88      88      11      00      00
VV      VV  88      88      11      00      00
VV      VV  88888888      111111      000000
VV      VV  88888888      111111      000000
VV      VV  88888888      111111      000000
VV      VV  88888888      111111      000000

```

```

LL          SSSSSSSS  TTTT'TTTT'TT
LL          SSSSSSSS  TTTT'TTTT'TT
SS          TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SSSSSS    TT
LL          SSSSSS    TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LLLLLLLLLLL SSSSSSSS  TT
LLLLLLLLLLL SSSSSSSS  TT

```

```

93      *$XXINI - INITIALIZE TASK POOL
94
95      : THIS ROUTINE IS CALLED TO DETERMINE THE STARTING ADDRESS AND LENGTH
96      : OF THE TASK POOL AND TO FORMAT THE POOL SO THAT IT MAY BE USED WITH
97      : THE "CORAL" ROUTINES.
98
99      : INPUTS:
100      :     NONE
101
102      : OUTPUTS:
103      :     $XXBUF=STARTING ADDRESS
104      :     $XXLEN=LENGTH
105      :     $XXAVL="CORAL" POINTER
106
107      : -
108
109      000000      .PSECT
110
111      000000      $XXINI::
112      000000      005767 000000'      TST      $XXBUF      ; HAS BUFFER ALREADY BEEN SETUP?
113      000004      001002      BNE      10$      ; IF NON-ZERO, YES
114      000006      10$:      CALL      XXSTR      ; GO DO THE WORK
115      000012      RETURN
116
117      000000      .PSECT      ..POOL,OV
118
119      000000      XXSTR:      CALL      $SAVAL      ; SAVE ALL REGISTERS
120      000004      012700 000000'      MOV      #XXSTR,R0      ; POOL START ADDRESS
121      000010      012701 000006'      MOV      #$XXAVL,R1      ; PLACE TO DEPOSIT IT
122      000014      064100      ADD      -(R1),R0      ; ROUND UP POOL START ADDRESS
123      000016      042100      BIC      (R1)+,R0
124      000020      010011      MOV      R0,(R1)      ; SET POOL START ADDRESS
125      000022      010067 000000'      MOV      R0,$XXBUF      ; TWICE
126      000026      000240      NOP      ; LEAVE ROOM FOR GTSK$S BUFFER
127      000030      000240      NOP      ; ...
128      000032      000240      NOP      ; ...
129      000034      000240      NOP      ; ...
130      000036      000240      NOP      ; ...
131      000040      000240      NOP      ; ...
132      000042      000240      NOP      ; ...
133      000044      000240      NOP      ; ...
134
135      ; DETERMINE SYSTEM TYPE ( RSX11D/RSX11M/RSX11S/IAS )
136
137      000046      GTSK$S R0      ; USE GTSK$
138      000056      103453      BCS      100$      ; DIRECTIVE NOT SUPPORTED ??
139      000060      016001 000034      MOV      G.TSSY(R0),R1      ; GET SYSTEM ID WORD
140      000064      020127 000006      CMP      R1,#6      ; IS THIS RSX-11M+ ?
141      000070      001412      BEQ      5$      ; IF EQ, YES
142      000072      020127 000001      CMP      R1,#1      ; IS THIS RSX11M ?
143      000076      001407      BEQ      5$      ; IF EQ, YES
144      000100      020127 000002      CMP      R1,#2      ; IS THIS RSX11S ?
145      000104      001404      BEQ      5$      ; IF EQ, YES
146
147      ; THEN THIS MUST BE RSX11D/IAS
148
149      000106      016002 000032      MOV      G.TSTS(R0),R2      ; GET TASK SIZE (IN WORDS)

```

214	000470			SWTCHO	#LOCBUF		: SWITCH OUTPUT BUFFER
215	000476			PUTAD	R4		: WRITE LISTHEAD OF BLOCK BEING DEALLOCATED
216	000506			RESMAP			: RESTORE PREVIOUS MAPPING
217	000512	010546		MOV	R5,-(SP)		: CALCULATE ADDRESS OF NEW BLOCK
218	000514	060116		ADD	R1,(SP)		
219	000516	020326		CMP	R3,(SP)+		: EQUAL TO NEXT IN CHAIN?
220	000520	001041		BNE	50\$: IF NE NO
221	000522			SAVMAP			: SAVE CURRENT MAPPING
222	000526			CEACCS	R3,R0		: CONVERT TO MAPPED ADDRESS
223	000536			SWTCHO	#LOCBF1		: SET UP LOCBF1 AS INPUT BUFFER
224	000544			GETAD	R0		: READ IN LISTHEAD
225	000554	012700	000012'	MOV	#LOCBF1,R0		: POINT TO LISTHEAD
226	000560	012046		MOV	(R0)+,-(SP)		: SAVE LINK WORD
227	000562	061001		ADD	(R0),R1		: MERGE TWO BLOCKS
228	000564	012667	177216	MOV	(SP)+,LOCBUF		: MOVE LINK WORD TO BLOCK RELEASED
229	000570			MAP	4(SP)		: GET MAPPING BIAS OF BLOCK BEING DEALLOCATED
230	000576	010167	177206	MOV	R1,LOCBUF+2		: STORE SIZE
231	000602			SWTCHO	#LOCBUF		: SWITCH OUTPUT BUFFER
232	000610			PUTAD	R4		: WRITE LISTHEAD OF BLOCK BEING DEALLOCATED
233	000620			RESMAP			: RESTORE PREVIOUS MAPPING
234	000624						: ADDRESS OF PREVIOUS BLOCK IS ON TOP OF STACK
235	000624	010567	177166	MOV	R5,LOCBF2		: ASSUME NO AGLOMERATION
236	000630			SWTCHO	#LOCBF2		: SWITCH OUTPUT BUFFER
237	000636			PUTAD	R2		: WRITE UPDATED LISTHEAD
238	000646	066716	177146	ADD	LOCBF2+2,(SP)		: CALCULATE ADDRESS OF NEXT BLOCK
239	000652	020526		CMP	R5,(SP)+		: EQUAL TO BLOCK BEING RELEASED?
240	000654	001026		BNE	60\$: IF NE NO
241	000656	066701	177136	ADD	LOCBF2+2,R1		: MERGE TWO BLOCKS
242	000662			SAVMAP			: SAVE CURRENT MAPPING
243	000666	016746	177114	MOV	LOCBUF,-(SP)		: SAVE LINK WORD
244	000672	012667	177120	MOV	(SP)+,LOCBF2		: MOVE LINK WORD TO PREVIOUS BLOCK
245	000676	012616		MOV	(SP)+,(SP)		: REPLACE BLOCK MAPPING
246	000700	010204		MOV	R2,R4		: SET NEW ADDRESS OF BLOCK
247							
248	000702			MAP	(SP)		: MAP TO CURRENTLY MERGED BLOCK
249	000706			SWTCHO	#LOCBF2		: SWITCH OUTPUT BUFFER
250	000714			PUTAD	R4,#2		: UPDATE NEXT AVAILABLE BLOCK POINTER
251							
252	000732			MAP	(SP)		: MAP TO BLOCK RELEASED
253	000736	010167	177056	MOV	R1,LOCBF2+2		: SET SIZE OF BLOCK RELEASED
254	000742	016767	177034	MOV	XAVL,\$XAVL		: UPDATE CEX COPY OF LISTHEAD
255	000750			SWTCHO	#LOCBF2+2		: SWITCH OUTPUT BUFFER
256	000756	052704	000002	ADD	#2,R4		: POINT TO LENGTH OF BLOCK
257	000762			PUTAD	R4,#2		: WRITE UPDATED LENGTH
258							
259	001000			SWTCHO	#SBFR		: SET UP SBFR AS DEFAULT INPUT
260	001006			SWTCHO	#SBFR		: AND OUTPUT BUFFER
261	001014	005726		TST	(SP)+		: CLEAN UP STACK
262	001016	012605		MOV	(SP)+,R5		: RESTORE REGISTERS
263	001020	012604		MOV	(SP)+,R4		: ...
264	001022			RETURN			: AND RETURN

VAC5

[illegible]

VAL T

E	10
F	10
G	10
H	10
I	10
J	10
K	10
L	10
M	10
N	10
B	11
C	11
D	11
E	11
F	11
G	11
H	11
I	11
J	11
K	11
L	11
M	12
N	12
B	12
C	12
D	12
E	12
F	12
G	12
H	12
I	12
J	12
K	12
L	12
M	13
N	13
B	13
C	13
D	13
E	13
F	13
G	13
H	13
I	13
J	13
K	13
L	13
M	14
N	14
B	14
C	14
D	14
E	14
F	14
G	14
H	14
I	14
J	14
K	14
L	14

VBUF

M	14
N	14
B	15
C	15
D	15
E	15
F	15
G	15
H	15
I	15
J	15
K	15
L	15
M	15
N	16
B	16
C	16
D	16
E	16
F	16
G	16
H	16
I	16
J	16
K	16
L	16
M	16

VCEX

```

5- 54  MACRO DEFINITIONS
6- 74  LOCAL DATA
7- 273 $SEX - TRANSFER COMM EXEC
9- 396 CEXYM - LOOK FOR CEX SYMBOLS
10- 510 NLBLK - SETUP NTL HOME BLOCK
11- 554 CEPR - INIT CETAB POINTERS

```

```
554 .SBTTL CEPTR - INIT CETAB POINTERS
555
556 ;+ CEPTR - INITIALIZE CETAB TABLE POINTERS
557
558 INPUTS:
559 R1=CETAB ALLOCATION ADDRESS
560
561 OUTPUTS:
562 R0,R1,R2=DESTROYED
563
564 CEPTR:
565 000746 MOV .PDVTB+10,$PDVTA ; INITIALIZE THE TABLE POINTERS
566 000746 016767 000644' 000000G MOV .SLTMB+10,$SLTMA ;
567 000754 016767 000656' 000000G MOV .LLCTB+10,$LLCTA ;
568 000762 016767 000670' 000000G MOV .SLTTOT+10,$SLTNM ; INITIALIZE THE NUMBER OF TABLE ENTRIES
569 000770 016767 000702' 000000G MOV PDVTOT+10,$PDVNM ;
570 000776 016767 000714' 000000G MOV ISLK+10,$CXOPT ; SET UP COMM EXEC OPTIONS
571 001004 016767 000620' 000000G
572
573 ; RELOCATE ADDRESSES
574 ADD R1,$PDVTA ; RELOCATABLE ADDRESSES
575 ADD R1,$SLTMA ;
576 ADD R1,$LLCTA ;
577
578 ; WRITE OUT THE INITIALIZED VALUES
579
580 001026 SWTCHO #$PDVTA ; POINT AT OUTPUT BUFFER
581 001034 PUTRC .PDVTA,#<$CCBNM-$PDVTA> ; WRITE OUT THE BUFFER
582 001054 016746 000000G MOV .PDVTA,-(SP) ; GET ADDRESS OF BEGINNING OF CETAB
583 001060 062716 000000C ADD #<$CXOPT-$PDVTA>,(SP) ; POINT TO CEXCM OPTIONS WORD
584 001064 SWTCHO #$CXOPT ; POINT AT OPTIONS WORD
585 001072 PUTRC (SP)+,#2 ; WRITE OUT OPTIONS WORD
586
587 ; SET UP MULTI-PROCESSOR CLOCK QUEUE ENTRIES
588
589 .IF DF,R$$MPL
590 MOV R0,-(SP)
591 MOV R1,-(SP)
592 MOV .MPRO,R0 ; GET NUMBER OF PROCESSORS
593 BLE 30$ ; BR IF SINGLE PROC SYSTEM
594 SWTCHO #TMPSTR ; RESET THE OUTPUT BUFFER
595 10$: MOV R0,R1 ; COPY PROCESSOR NUMBER
596 DEC R1 ; OFFSET BY ZERO
597 ASL R1 ; MAKE INTO A WORD TABLE INDEX
598 ADD .K6TAB,R1 ; GE: ADDRESS IN CPU AREA TABLE
599 SWTCHI #SRBLCK ; READ INTO THE APR5 BIAS WORD
600 GETRV R1,#2 ; GE: THE CPU AREA BIAS
601 CLR TMPSTR ; MAKE A NULL
602 MOV .M100Q,R1 ; GET THE CLOCK QUEUE ADDRESS
603 BIS #120000,R1 ; USE APR5
604 PUTAD R1,#2 ; INITIALIZE THE 100 MSEC TIMER QUEUE
605 MOV .M100Q,R1 ; SET SECOND WORD OF LIST HEAD
606 MOV R1,TMPSTR ;
607 ADD #2,R1 ;
608 BIS #120000,R1 ;
609 PUTAD R1 ;
610 DEC R0 ; NEXT PROCESSOR
```

LR.EFN
LR.LIN
LR.PRM
LR.PRO
LR.STS
LR.TCB
LR.UNT
LS.AST
LS.CEX
LS.CIR
LS.CXO
LS.CX5
LS.DLM
LS.ECH
LS.FDX
LS.HDX
LS.LIN
LS.LMC
LS.NTI
LS.ON
LS.OPT
LS.PRO
LS.PWF
LS.TOP
LS.UNF
LS.X25
LS.110
LX.CEX
LX.LIN
LX.PRO
MSGB
MSGOUT
M\$\$MGE
NLBLK
NTINAM
PDVTOT
P.BLKS
P.REL
RJN
RUNNTI
R\$\$EIS
R\$\$MPL
R\$\$110
SLTTOT
STBERR
STEXT
SYSFDB

VCEX CREFATED BY MACRO ON 29-JUN-85 AT 02:14 PAGE 2 B 3

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
LR.EFN	000004	#5-65
LR.LIN	000002	#5-65
LR.PRM	000030	#5-65
LR.PRO	000002	#5-65
LR.STS	000000	#5-65 14-792 *14-799 *14-813 *14-814
LR.TCB	000002	#5-65
LR.UNT	000005	#5-65
LS.AST	= 100000	#5-65
LS.CEX	= 000001	#5-65 14-792 14-799 14-813
LS.CIR	= 000010	#5-65
LS.CXO	= 010000	#5-65
LS.CX5	= 000400	#5-65
LS.DLM	= 004000	#5-65
LS.ECH	= 001000	#5-65
LS.FDX	= 004000	#5-65
LS.HDX	= 010000	#5-65
LS.LIN	= 030003	#5-65
LS.LMC	= 000007	#5-65
LS.NTI	= 004000	#5-65
LS.ON	= 000011	#5-65
LS.OPT	= 000400	#5-65
LS.PRO	= 000002	#5-65
LS.PWF	= 040000	#5-65
LS.TOP	= 002000	#5-65
LS.UNF	= 020000	#5-65 14-814
LS.X25	= 000012	#5-65
LS.11D	= 004000	#5-65
LX.CEX	= 000004	#5-65
LX.LIN	= 000006	#5-65
LX.PRO	= 000005	#5-65
MSGB	000254 R	#6-104 15-865
MSGOUT	002070 R	13-721 14-741 14-753 14-781 14-843 #15-864
MSGMGE	= 000000	6-248 6-260
NBLK	000666 R	8-357 #10-522
NTINAM	000430 R	#6-154 14-737
PDVTOT	000704 R	6-225 #6-228 11-569
P.BLKS	000016	7-320
P.REL	000014	7-321
RJN	001206 R	#13-669 14-841
RUNNTI	001504 R	7-297 #14-736
RSEIS	= *****	7-322
RSMPL	= *****	6-120 6-140 6-160 7-302 7-311 8-361 9-413 9-437 9-458
		10-525 10-537 11-589 14-747 14-765 14-771 14-791 14-809 14-828
R\$11D	= *****	7-322
SLTTOT	000672 R	6-222 #6-225 11-568
STBERR	000660 R	9-431 9-449 9-456 #9-507
STBEXT	000402 R	#6-133 9-420 9-445 9-452
SYSFDB	= ***** GX	*7-318 *7-325 *8-350 *8-363 *8-379 *11-580 *11-581 *11-584 *11-585
		*14-848 *12-636 *13-685 *13-692 *13-711 *13-713 *14-786 *14-789 *14-837
TICKS	= 000001	#6-86 14-840
TSKERR	000452 R	7-299 7-315 8-356 8-367 8-375 #8-392

VCFG -
\$NLCFG

247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304

```

247          .SBTTL $NLCFG - CONFIG FILE SCAN
248          *
249          $NLCFG - PROCESS CONFIG FILE FOR MISSING PARAMETERS
250
251          INPUTS:
252          $MISS=MISSING PARAMETERS MASK
253          $SLTA=SLT ADDRESS
254
255          OUTPUTS:
256          C-BIT=SUCCESS/FAILURE
257
258          $NLCFG::
259          .MCALL VNPBPT
260          VNPBPT <VCFG>
261          CALL $CLSAV          ; SAVE GCL CONTEXT
262          MOVB #CL.OPN,$CLFLG ; SET UP NECESSARY CONTEXT
263          MOV SP,$PSAV        ; SAVE SP FOR ERROR EXIT
264          CLR SCNT            ; CLEAR LOCAL COUNTERS
265          CLR UCNT
266          MOV #NAME,RO        ; CONVERT DDM NAME TO ASCII
267          MOV $NAME,R1        ; AND STORE IN "NAME"
268          CALL $CSTA
269          MOV $SLTA,RO        ; GET SLT ADDRESS
270          MOVB L.CTL(RO),CTL  ; COPY CONTROLLER NUMBER
271          MOVB L.UNT(RO),UNT  ; AND UNIT NUMBER
272          CALL OPEN          ; OPEN THE CONFIG FILE
273
274          BIT #MS.CIR,$MISS   ; ARE CIRCUIT PARAMETERS MISSING?
275          BEQ 80$             ; BR IF NO
276
277          ;
278          ; PVC$DF will be ignored for gateway systems, since the X25 circuit
279          ; block must be allocated from a private PSI pool which is not
280          ; available until runtime.
281          ;
282
283          10$: CALL READ      ; READ NEXT RECORD
284          BCC 20$            ; BR IF OK
285          JMP 111$           ; IF CS, EOF
286          20$: CALL QSLT     ; IS IT SLT$DF?
287          BCS 10$            ; BR IF NO - KEEP LOOKING
288          30$: CALL READ     ; GET NEXT RECORD
289          BCC 40$            ; BR IF OK
290          JMP 141$           ; IF CS, EOF
291          40$: CALL QSTA     ; IS IT STAS$DF?
292          BCC 30$            ; IF CC, YES - LOOK FOR MORE OF THEM
293          50$: CALL READ     ; READ NEXT RECORD
294          BCC 60$            ; BR IF OK
295          JMP 181$           ; BR IF EOF
296          60$: CALL $QSV     ; IS IT SVC$DF?
297          BCC 70$            ; BR IF YES
298          CALL $QX3P         ; IS IT X3P$DF?
299          BCC 50$            ; BR IF YES
300          CALL $DTE         ; IS IT DTE$DF?
301          BCC 50$            ; BR IF YES
302          BIT #PF.PSV,$PFLAG ; Ignore PVC$DF for gateway systems
303          BNE 50$

```

[TD001]
[TD001]
[TD001]
[TD001]
[TD001]

[TD001]
[TD001]

VCFG -
TPARS

1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275

B 6

```

1260
1261
1262
1263
1264 002364 005767 000000G
1265 002370 001015
1266 002372 016700 000150'
1267 002376 005300
1268 002400 006300
1269 002402 016760 000000G 000000G
1270 002410 006367 000000G
1271 002414 016760 000000G 000000G
1272 002422
1273
1274 002424
1275

; HELLO/LISTEN TIMER
ST.HTM: TST .PNUMH ; DOUBLE WORD VALUF?
        BNE 101$ ; BR IF YES - ERROR
        MOV SCNT,RO ; GET CURRENT STATION COUNT
        DEC RO ; REALLY LAST ONE
        ASL RO ; MAKE IT A WORD OFFSET
        MOV .PNUMB,$$HTIM(RO) ; SAVE HELLO TIMER
        ASL .PNUMB ; MULTIPLY BY 2
        MOV .PNUMB,$$LTIM(RO) ; SAVE LISTEN TIMER
        RETURN
101$: MSG$R Y4 ; ILLEGAL HELLO TIMER

```

P\$RTIM
 P\$STA1
 P\$STA2
 P\$TIM
 P\$TSC1
 P\$TSC2
 P\$TYP
 P\$P45
 P\$WRD
 P.EXT
 P.INC
 QADD
 QCNT
 QDDM
 QFEA
 QLLC
 QPLT
 QSLT
 QSTA
 QUNT
 Q\$OPT
 READ
 RF.CTL
 RF.LD1
 RF.LD2
 RF.TIM
 RF.TMO
 RF.WFC
 RF.WTD
 RF.WTM
 RF.WTS
 RTSPC
 R\$DER
 R\$K11
 R\$SND
 R\$11M
 SCHEL
 SCHELO
 SCNT
 SCTIM
 SETTMO
 SLTD
 SPACE
 . ABS.
 DATA
 \$STATE
 \$KTAB
 \$KSTR
 Errors

*** As

Work
Work

C 6

```

Symbol table
P$RTIM 000003
P$STA1 000022
P$STA2 000023
P$TIM 000010
P$TSC1 000026
P$TIS1 000024
P$TYP 000001
P$P45= 000000
P$WRD= 000000
P.EXT 001652R
P.INC 001576R
QADD 001124R
QCNT 001140R
QDDM 001132R
QFEA 001154R
QLLC 001162R
QPLT 001064R
QSLT 001072R
QSTA 001116R
QUNT 001146R
Q$OPT= 000010
READ 000772R
RF.CTL= 000003
RF.LD1= 000040
RF.LD2= 000100
RF.TIM= 177400
RF.TMO= 000400
RF.WFC= 000200
RF.WTD= 000020
RF.WTM= 000030
RF.WTS= 000010
RTSPC 001330RG
R$DER= 000000
R$K11= 000001
R$SND= 000000
R$TIM= 000000
SCHEL 000112R
SCHELO 000102R
SCNT 000150R
SCTIM 000070R
SETTMO 003322P
SLTD 000000R
SPACE = 000040

SPEEDT 000156R
SPFILL 001452RG
SPSAV 002702R
STADF 000244R
STADF1 000272R
STADF2 000304R
STORID 001332RG
ST.APR 002332R
ST.CST 002252R
ST.HTM 002364R
ST.NUM 002164R
SYNERR 002704RG
SYSFDB= ***** GX
$$$WRG= 000000
$$$YSZ= 007600
S.BRA 001752R
S.COST= ***** GX
S.CTIM 002014R
S.CTL 001706R
S.HTIM 002040R
S.NAME 001512R
S.PRI 001762R
S.UNT 001730R
T$KMG= 000000
T$MIN= 000000
UCNT 000152R
UMRFL 000154R
UNT 000016R
UNTD 000514R
U.CHAO 002762R
U.CHAT 003066R
U.CST 003602R
U.DPR 003634R
U.HTIM 003644R
U.LTIM 003670R
U.PECH 003140R
U.SCSR 003204R
U.UNT 002716R
U.XCSR 003536R
V$RCV= 100000
V$XMT= 040000
V$FLG 000000
V$LEN 000022

002 V$RCV 000002
V$XMT 000012
002 V$CTR= 001000
003 WND$MX= 000177
003 X$SDBT= 000000
003 Z.NAM= ***** GX
$ALPHA= 000022
$ANY= 000020
$AZFS = ***** GX
$BFR = ***** GX
$BLANK= 000006
002 $CEACC= ***** GX
$CLBUF= ***** GX
$CLDEQ= ***** GX
$CLFLG= ***** GX
$CLIOS= ***** GX
$CLQUE= ***** GX
$CLSAV= ***** GX
$CLSBK= ***** GX
$CLSIZ= ***** GX
$CTIM = ***** GX
$CSTA = ***** GX
$DFUIC= ***** GX
$DIGIT= 000024
$DIV = ***** GX
002 $DNUMB= 000014
002 $EOS = 000012
002 $ERRYA 001702R
002 $ERRYB 001734R
003 $ERRYP 001770R
$ERRYQ 002012R
$ERRYR 002052R
$ERRYRS 002106R
$ERRRZ 001630R
$ERR1A 000216R
$ERR1B 000246R
$ERR1C 000306R
$ERR1D 000340R
$ERR1E 000376R
$ERR1F 000424R
$ERR1G 000460R
$ERR1H 000506R
$ERR1J 000544R

$ERR1K 000600R
$ERR1L 000640R
$ERR1M 000700R
$ERR1N 000744R
$ERR1P 001004R
$ERR1Q 001042R
$ERR1R 001100R
$ERR1S 001130R
$ERR1T 001160RG
$ERR1U 001206R
$ERR1V 001246R
$ERR1W 001306R
$ERR1X 001346R
$ERR1Y 001406R
$ERR1Z 001456R
$ERR2A 001516R
$ERR2B 001566R
$ERR2C 002312RG
$ERR2F 002346RG
$ERR2I 002402RG
$ERR3A 002152R
$ERR3B 002210R
$ERR3C 002246R
$EXIT = 000000
$FAIL = 177777
$FEATR= ***** GX
$FLAGS= ***** GX
$GCLEP= ***** GX
$GETAD= ***** GX
$GETRV= ***** GX
$LAMDA= 000000
$MISS = ***** GX
$MXVCT= ***** GX
$NAME = ***** GX
$NLCFG 000000RG
$NUMBR= 000002
$PDVNM= ***** GX
$PDVTA= ***** GX
$PFLAG= ***** GX
$QDTE = ***** GX
$QPVIC = ***** GX
$QSVIC = ***** GX

002 $QX2P = ***** GX
002 $QX3P = ***** GX
002 $RADDR= ***** GX
002 $RAD50= 000016
002 $RDBSZ= ***** GX
002 $REP.N= ***** GX
002 $SLTA = ***** GX
002 $STRNG= 000004
002 $SUBXP= 000010
002 $TIOUT= ***** GX
002 $XLINK= ***** GX
002 $SAPR = ***** GX
002 $SCSR = ***** GX
002 $SDCHA= ***** GX
002 $SDEV = ***** GX
002 $SDPR = ***** GX
002 $SEXT = ***** GX
002 $SHTIM= ***** GX
002 $SINC = ***** GX
002 $SLAD= ***** GX
002 $SLTIM= ***** GX
002 $SMTPE= ***** GX
002 $SMUX = ***** GX
002 $SNBRA= ***** GX
002 $SPCHA= ***** GX
002 $SPRI = ***** GX
002 $SRPRI= ***** GX
002 $SCSR= ***** GX
002 $SNUM= ***** GX
002 $SVE.T= ***** GX
002 $$$FLG= 177777
002 $$$KEY= 000016
002 $$$STA= 000000
002 $$$TMP= 000137R
002 .MGME= ***** GX
002 .PCHAR= ***** GX
002 .PNUMB= ***** GX
002 .PNUMH= ***** GX
002 .PSTCN= ***** GX
002 .PSTPT= ***** GX
002 .TPARS= ***** GX
002 .SCSR= ***** GX

```

```

. ABS. 000124 000 (RW,I,GBL,ABS,OVR)
      004262 001 (RW,I,LCL,REL,CON)
DATA 002706 002 (RW,D,LCL,REL,CON)
$STATE 001056 003 (RW,D,LCL,REL,CON)
$XTAB 000036 004 (RW,D,LCL,REL,CON)
$KSTR 000142 005 (RW,D,LCL,REL,CON)
Errors detected: 0

```

*** Assembler statistics

work file reads: 24
work file writes: 32

```

59      .SBTTL  MACRO DEFINITIONS
60      ;***
61      ; LIBRARY MACROS
62      ;***
63      .MCALL  ASL$,EMSG$,ISTAT$,NTLERS$,STATES$,TRANS$
64      .MCALL  SAYRG,RESRG,SLTDF$,CEACC$,GETAD,GETRV
65      .MCALL  VNPDBG
66
67      SLTDF$                                ; DEFINE SLT OFFSETS AND SYMBOLS
68
69      ;***
70      ; LOCAL MACROS
71      ;***
72
73      ; REJECT TPA4S TRANSITION
74      ;
75      .MACRO  REJ$
76      ADD    #2,(SP)                        ; RETURN TO CALLER+2
77      CLR    SYNERR                        ; INDICATE NO SYNTAX ERROR
78      .ENDM  REJ$
  
```

;[TD001]
 ;**~1

```

518 .SBTTL COMMON ACTION ROUTINES - LINE-ID
519
520 .ENABL LSB
521
522 : DEVICE NAME
523
524 000652 022767 000003 000000G DEVNAM::CMP #3,.PSTCN ; VALID DEVICE NAME?
525 000660 103425 BLO 101$ ; BR IF NO
526 000662 016700 000000G MOV .PSTPT,RO ; GET ADDRESS OF FIRST CHARACTER
527 000666 CALL $CAT5 ; CONVERT TO RAD50
528 000672 010167 000346' MOV R1,LINNAM ; SAVE DEVICENAME
529 000676 RETURN
530
531 : DEVICE CONTROLLER NUMBER
532
533 000700 005767 000000G DEVCTL::IST .PNUMH ; LEGAL CONTROLLER NUMBER VALUE?
534 000704 001013 BNE 101$ ; BR IF NO
535 000706 016767 000000G 000350' MOV .PNUMB,LINCTL ; SAVE CONTROLLER NUMBER
536 000714 RETURN
537
538 : DEVICE UNIT NUMBER
539
540 000716 005767 000000G DEVUNT::TST .PNUMH ; LEGAL UNIT NUMBER VALUE?
541 000722 001004 BNE 101$ ; BR IF NO
542 000724 116767 000000G 000351' MOV .PNUMB,LINUNT ; SAVE UNIT NUMBER
543 000732 RETURN
544
545 : ERRORS
546
547 000734 101$: MSG$R 06 ; ILLEGAL LINE-ID
548 .DSABL LSB
549
550 : CHECK FOR LEGAL BLOCK SIZE VALUE
551
552 000742 005767 000000G CHKBLK::TST .PNUMH ; DOUBLE WORD VALUE?
553 000746 001012 BNE 10$ ; BR IF YES - ERROR
554 000750 016700 000000G MOV .PNUMB,RO ; GET BLOCK SIZE VALUE
555 000754 022700 000040 CMP #BLKSMN,RO ; IS BLOCK SIZE VALUE WITHIN RANGE?
556 000760 101005 BHI 10$ ; BR IF NO - ERROR
557 000762 022700 002000 CMP #BLKSMX,RO
558 000766 103402 BLO 10$ ; BR IF NO - ERROR
559 000770 000241 CLC ; INDICATE LEGAL BLOCK SIZE VALUE
560 000772 000401 BR 20$
561 000774 000261 10$: SEC ; INDICATE ILLEGAL VALUE
562 000776 20$: RETURN
563
564 : CHECK FOR LEGAL WINDOW SIZE VALUE
565
566 001000 005767 000000G CHKWND::TST .PNUMH ; DOUBLE WORD VALUE?
567 001004 001010 BNE 10$ ; BR IF YES - ERROR
568 001006 016700 000000G MOV .PNUMB,RO ; GET BLOCK SIZE VALUE
569 001012 003405 BLE 10$ ; BR IF ILLEGAL VALUE
570 001014 022700 000177 CMP #WNDSMX,RO ; IS VALUE IN RANGE?
571 001020 103402 BLO 10$ ; BR IF NO - ERROR
572 001022 000241 CLC ; INDICATE LEGAL WINDOW SIZE VALUE
573 001024 000401 BR 20$
574 001026 000261 10$: SEC ; INDICATE ILLEGAL VALUE

```


ASSCHK= 000000	ERR20 000024	F3.TCM= 002000	L\$BROB 000364	R\$\$SND= 000000
ASSCPS= 000000	ERR22 000026	F3.UDS= 000020	L\$BROL 000366	R\$\$11M= 000000
ASSPRI= 000000	ERR24 000030	F3.WAT= 010000	L\$BSA 000010	SISDRO= 172200
AS\$TRP= 000000	ERR26 000032	F3.XHR= 000100	L\$BSEG 000026	SRO = 177572
CMODE = 140000	ERR28 000034	F3.11S= 000400	L\$BSGL 000354	SR3 = 172516
C\$SCKP= 000000	ERR30 000036	G\$STPP= 000000	L\$BSYS 000025	SWR = 177570
C\$SORE= 000400	ERR32 000040	G\$STSS= 000000	L\$BTSK 000000	SYSFDB= ***** GX
C\$SRSH= 177564	ERR34 000042	G\$STTK= 000000	L\$BWND 000024	S\$SWRG= 000000
DSKBUF 000000R	ERR36 000044	G\$SWRD= 000000	L\$BXFR 000350	S\$YSZ= 007600
DSAMXC 000072	ERR4 000004	HF.CIS= 000200	L\$SASG= 000000	TPS = 177564
DSAMXH 000074	ERR6 000006	HF.EIS= 000002	L\$SDRV= 000000	TS\$ACP= 020000
DSANN 000000	ERR8 000010	HF.FPP= 100000	L\$SP11= 000001	TS\$CHK= 000100
DSBRPR 000102	E\$XPR= 000000	HF.QB= 000004	L\$S11R= 000000	TS\$CMP= 000200
DSBRTM 000100	FE.CAL= 000040	HF.UBM= 000001	MPAR = 172100	TS\$IOP= 000020
DSDEL 000045	FE.CEX= 020000	I\$SAR= 000000	MPCSR = 177746	TS\$NEW= 000001
DSDELW 000046	FE.DRV= 000010	I\$SRDN= 000000	M\$SARB= 000124	TS\$JHD= 040000
DSEND = 000104	FE.DYM= 010000	KDSARO= 172360	M\$SARX= 000000	TS\$NSD= 002000
DSFNB 000034	FE.EXP= 000200	KDSDR0= 172320	M\$SFCS= 000000	TS\$NXH= 000002
DSHIOR 000024	FE.EXT= 000001	KJNARO= 172340	M\$SMGE= 000000	TS\$PIC= 100000
DSHOST 000022	FE.EVX= 000004	KJNARS= 172352	M\$SNET= 000000	TS\$PMO= 010000
DSINAC 000044	FE.FDT= 002000	KJNARG= 172354	M\$SOVR= 000000	TS\$PRV= 000400
DSINCT 000042	FE.LS1= 000400	KJNAR7= 172356	N\$SACC= 000001	TS\$RES= 000040
DSIPL 000051	FE.MUP= 000002	KISARO= 172340	N\$SBUF= 000001	TS\$SLV= 004000
DSLID 000020	FE.MXT= 040000	KISARS= 172352	N\$SLDV= 000001	TS\$SUP= 000010
DSLNAM 000006	FE.NLG= 100000	KISARG= 172354	N\$SMCP= 000001	TS\$XHR= 000004
DSLNUM 000014	FE.OFF= 001000	KISAR7= 172356	N\$SMML= 000001	TS\$KMG= 000000
DSLST 000047	FE.PKT= 000100	KISDR0= 172300	N\$SMOV= 000010	TS\$MIN= 000000
DSMAXC 000064	FE.PLA= 000020	KISDR6= 172314	N\$SNCT= 000001	T2\$CL1= 000001
DSMAXH 000066	FE.X25= 004000	KISDR7= 172316	N\$SPEM= 000001	T2\$FMP= 000002
DSMAXV 000070	F\$SVL= 000001	K\$SCNT= 177546	PIRQ = 177772	UBMPR = 170200
DSMLL 000040	F.RSIZ= ***** GX	K\$SCSR= 177546	PMODE = 030000	UDSARO= 177660
DSMNOD 000041	F2.ACN= 000020	K\$SLDC= 000000	PR.LBL 000076R	UDSDRO= 177620
DSNA 000062	F2.AHR= 010000	K\$STPS= 000074	PR.XFR 000272R	UISARO= 177640
DSNBEA 000036	F2.DAS= 000001	L\$SREV= 000400	PRO = 000000	UISAR4= 177650
DSNBRA 000054	F2.DPR= 000400	LD\$ACC= 100000	PR1 = 000040	UISAR5= 177652
DSNEND= 000054	F2.EVT= 000010	LD\$CLS= 020000	PR4 = 000200	UISAR6= 177654
DSNLN 000030	F2.GGF= 002000	LD\$LP = 000000	PR5 = 000240	UISAR7= 177656
DSNN 000050	F2.IRR= 001000	LD\$REL= 000004	PR6 = 000300	UISDR0= 177600
DSOUT 000043	F2.LIB= 000002	LD\$RSV= 040000	PR7 = 000340	UISDR4= 177610
DSRETF 000050	F2.MP = 000004	LD\$SUP= 000010	PS = 177776	UISDR5= 177612
DSRNN 000002	F2.POL= 000100	L\$BASG 001000	P\$P45= 000000	UISDR6= 177614
DSRTMR 000076	F2.RAS= 004000	L\$BLK 000360	P\$WRD= 000000	UISDR7= 177616
DSSEG 000036	F2.RBN= 020000	L\$BDAT 000032	Q\$DOPT= 000010	V\$CTR= 001000
DSER 000032	F2.SDW= 000040	L\$BEXT 000352	R\$LDAT 000026	X\$SDBT= 000000
DSQRL 000052	F2.STP= 100000	L\$BFLG 000030	R\$FLG 000024	\$ALL15= ***** GX
DS\$BUG= 177574	F2.SWP= 040000	L\$BFL2 000774	R\$LHGV 000006	\$ALL16= ***** GX
DS\$ISK= 000000	F2.WND= 000200	L\$BHGV 000012	R\$LLDZ 000012	\$BFR = ***** GX
DS\$11= 000001	F3.AST= 000200	L\$BHRB 000356	R\$LMXV 000010	\$CLDEQ= ***** GX
DS\$YNC= 000000	F3.CLI= 001000	L\$BLDZ 000016	R\$LMXZ 000014	\$CLIOS= ***** GX
DS\$YNM= 000000	F3.CRA= 000001	L\$BLIB 000040	R\$LNAM 000000	\$CLOPE= ***** GX
ERROR 000000R	F3.EIS= 000004	L\$BLRL 000776	R\$LOFF 000016	\$CTBIO 000000RG
ERR10 000012	F3.NWK= 000002	L\$BLUN 000362	R\$LSA 000004	\$DECHM 000416RG
ERR12 000014	F3.PMN= 004000	L\$BMXV 000014	R\$LSEG 000022	\$DEOPT= ***** GX
ERR14 000016	F3.PRO= 000040	L\$BMXZ 000020	R\$LSIZ 000034	\$IOSB = ***** GX
ERR16 000020	F3.RLK= 020000	L\$BOFF 000022	R\$LWND 000020	\$LBN = ***** GX
ERR18 000022	F3.SHF= 040000	L\$BPAR 000004	R\$DER= 000000	\$LBXL = 000340
ERR2 000002	F3.STM= 000010	L\$BPR1 000346	R\$K11= 000001	\$LEN = ***** GX

VCLQRM - REMOVE ENTRIES FROM CL MACRO V05.03b Saturday 29-Jun-85 02:18 Page 8-2
Symbol table

000306 001 (RW,I,LCL,REL,CON)
Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 13275 Words (52 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:11.52

SY:VCLQRM.V2,[132,134]VCLQRM/CR/~SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VCLQRM


```

188 .SBTTL $NLDEV - SETUP DATA STRUCTURES
189
190 *
191 $NLDEV - SETUP RSX11M DEVICE DATA STRUCTURES
192
193 THIS ROUTINE IS CALLED WHEN DOING A "SET PROCESS" OF AN LLC IF THE LLC$DF
194 MACRO IN CETAB INCLUDED A DEVICE NAME (SIXTH MACRO ARGUMENT). IF THE DEVICE
195 DATA STRUCTURES ALREADY EXIST, THE PCB ADDRESS IS COPIED FROM THE PDV INTO
196 THE DCB. IF THE DATA STRUCTURES DO NOT EXIST, THE FILE [131,54]XXXTAB.TSK
197 IS ASSUMED TO BE AN IMAGE COPY OF THE DEVICE TABLES.
198
199 INPUTS:
200 $$$DEV=2 CHARACTER DEVICE NAME (ASCII)
201 $PDVA=PDV ADDRESS
202 $NAME=PROCESS NAME (RAD50)
203
204 OUTPUTS:
205 C-BIT=SUCCESS/FAILURE
206
207 $NLDEV::
208 MCALL VNPBPT
209 VNPBPT VDEV
210 SWITCH #DEVBUF ; DEFAULT TO DEVBUF
211 SWITCH #DEVBUF ; DEFAULT TO DEVBUF
212 CALL FIND ; SEE IF DATA STRUCTURES ALREADY EXIST
213 BCS 20$ ; IF CS, DRIVER ALREADY RESIDENT!
214 CALL DVSYM ; LOOK FOR SPECIAL SYMBOLS
215 BCS 20$ ; IF CS, STB FILE ERROR
216 TST FOUND ; DO STRUCTURES ALREADY EXIST?
217 BNE 10$ ; IF NE, YES
218 CALL DVLBL ; OPEN IMAGE FILE, READ LABEL BLOCK
219 BCS 20$ ; IF CS, FAILURE
220 CALL DVALL ; ALLOCATE SPACE FROM THE POOL
221 BCS 20$ ; IF CS, FAILURE
222 CALL DVXFR ; TRANSFER STRUCTURES INTO CORE
223 BCS 20$ ; IF CS, FAILURE
224 CALL $CLDEQ ; CLOSE IMAGE FILE
225 MOV $PDVA,R0 ; GET PDV ADDRESS
226 MOV DCD,R3 ; AND DCB ADDRESS
227 GETRV R3,#D.PCB+2 ; READ IN DCB
228 MOV $$TBL,D.DSP+DEVBUF ; COPY DISPATCH TABLE ADDRESS
229 BEQ 101$ ; IF EQ, ERROR
230 MOV Z.PCB(R0),D.PCB+DEVBUF ; COPY PCB ADDRESS
231 MOV D.UCB+DEVBUF,R0 ; SAVE UCB ADDRESS
232 MOV D.UCBL+DEVBUF,R1 ; AND UCB LENGTH
233 PUTRC ; REWRITE DCB
234 GETRV R0,R1 ; READ IN NEXT UCB
235 MOV $FEATR,U.CW3+DEVBUF ; STORE THE FEATURES WORD
236
237 .IF DF R$$MPL ; [MP01]
238 SWITCH #BIAS ; SET UP OUTPUT BUFFER ADDRESS ; [MP01]
239 MOV U.SCB+DEVBUF,-(SP) ; COPY SCB ADDRESS ; [MP01]
240 ADD .SKSS,(SP) ; POINT TO S.SK5 OFFSET ; [MP01]
241 PUTRC (SP),#2 ; UPDATE S.KSS ; [MP01]
242 TST (SP)+ ; CLEANUP STACK ; [MP01]
243 SWITCH #DEVBUF ; RESET OUTPUT BUFFER ; [MP01]
244 CMP #^RDLX,$NAME ; DLX ? ; [MP01]

```

```

696 .SBTTL REP4A - REPLACE XXXTAB NAME
697 .SBTTL REP4A1 - REPLACE XXX NAME
698
699 ;+
700 REP4A - REPLACE XXXTAB NAME AND SYMBOL NAME
701 REP4A1 - REPLACE XXX NAME AND SYMBOL NAME
702
703 INPUTS:
704 R1=SYMBOL LOOKUP CONTROL BLOCK ADDRESS
705
706 OUTPUTS:
707 R1=DESTROYED
708
709 REP4A: MOV R1, -(SP) ; SAVE R1
710 CALL $REP.P ; REPLACE XXXTAB NAME
711 BR REP4A2
712 REP4A1: MOV R1, -(SP) ; SAVE R1
713 10$: MOV (R3)+, (R4) ; COPY FORMAT STRING
714 CMPB #'*, (R4)+ ; LOOKING FOR FIRST ASTERISK
715 BNE 10$ ;
716 DEC R4 ; SKIP THE ASTERISK
717 MOV R4, R0 ; COPY BUFFER ADDRESS
718 MOV $NAME, R1 ; GET LLC NAME
719 CALL $CSTA ; CONVERT TO ASCII
720 20$: CMPB #' ', -(R0) ; DISCARD TRAILING SPACES
721 BEQ 20$ ;
722 INC R0 ;
723 MOV R0, R4 ; SET NEW BUFFER ADDRESS
724 REP4A2: MOV (SP)+, R1 ; RESTORE R1
725 10$: MOV (R3)+, (R4) ; COPY FORMAT STRING
726 CMPB #'*, (R4)+ ; LOOKING FOR THE SECOND ASTERISK
727 BNE 10$ ;
728 MOV R4, R0 ; COPY BUFFER POINTER
729 MOV #'", -(R4) ; DEPOSIT A DOUBLE QUOTE
730 MOV 6(R1), -(SP) ; SAVE SECOND HALF OF NAME
731 MOV 4(R1), R1 ; GET FIRST HALF OF NAME
732 CALL $CSTA ; CONVERT TO ASCII
733 MOV (SP)+, R1 ; RESTORE SECOND HALF OF NAME
734 20$: CALL $CSTA ; CONVERT TO ASCII
735 CMPB #' ', -(R0) ; DISCARD TRAILING BLANKS
736 BEQ 20$ ;
737 INC R0 ;
738 MOV (R0)+, R1 ; DEPOSIT ANOTHER DOUBLE QUOTE
739 MOV R0, R4 ; SET NEW BUFFER POINTER
740 RETURN
  
```

```

105          .SBTTL  LOCAL DATA
106
107          ;+
108          ; LOCAL DATA
109          ; -
110
111          .ENABL  LC
112
113          ; ASCIZ STRINGS FOR COMMAND TYPES
114
115          ;
116 000000      145      166      145  EVENT: .ASCIZ  /events/
117 000007      163      165      155  SUMARY: .ASCIZ  /summary/
118 000017      143      150      141  CHARAC: .ASCIZ  /characteristics/
119
120          ;
121          ; ASCIZ STRINGS FOR OBJECT HEADER MESSAGE
122          ;
123 000037      113      156      157  OKNOWN: .ASCIZ  /known objects/
124 000055      117      142      152  OSPEC:  .ASCIZ  /Object/
125
126          ;
127          ; ASCIZ STRINGS FOR OBJECT ACCESS LEVELS, PROCESS, NODE AND LINE STATES
128          ;
129 000064      117      156      000  ON:    .ASCIZ  /On/
130 000067      111      156      163  INSPCT: .ASCIZ  /Inspect/
131 000077      117      146      146  CFF:   .ASCIZ  /Off/
132 000103      123      150      165  SHUT:  .ASCIZ  /Shut/
133
134          ;
135          ; ASCIZ STRINGS FOR USER INFORMATION
136          ;
137 000110      104      145      146  DEFAULT: .ASCIZ  /Default/
138 000120      114      157      147  LOGIN:  .ASCIZ  /Login/
139
140          ;
141          ; ASCIZ STRING FOR COPIES INFORMATION
142          ;
143 000126      123      151      156  SINGLE: .ASCIZ  /Single/
144
145          ;
146          ; ASCIZ STRING FOR ALIAS HEADER MESSAGE
147          ;
148 000135      101      154      154  AALL:   .ASCIZ  /All aliases/
149 000151      113      156      157  AKNOWN: .ASCIZ  /Known aliases/
150 000167      101      154      151  ASPEC:  .ASCIZ  /Alias/
151
152          ;
153          ; ASCIZ STRING FOR PROCESS HEADER MESSAGE
154          ;
155 000175      101      143      164  PACTIV: .ASCIZ  /Active processes/
156 000216      113      156      157  PKNOWN: .ASCIZ  /Known processes/
157 000236      120      162      157  PSPEC:  .ASCIZ  /Process/
158
159          ;
160          ; ASCIZ STRING FOR PROCESS AND LINE STATES
161          ;

```

VDIS - VNP SHOW COMMANDS
\$DLOG - SHOW LOGGING

MACRO V05.03b Monday 15-Jul-85 12:14 Page 12

.SBTTL \$DLOG - SHOW LOGGING

```

+
$DLOG - SHOW ACTIVE LOGGING
      SHOW KNOWN LOGGING
      SHOW LOGGING MONITOR
      SHOW LOGGING CONSOLE
      SHOW LOGGING FILE

```

INPUTS: NONE

OUTPUTS: LOGGING INFORMATION IS DISPLAYED ON USER'S TERMINAL

ALL REGISTERS ARE DESTROYED

-

```

645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663 003446 012700 021274 $DLOG:: MOV #^REVL,R0 : GET EVL PROCESS NAME
664 003452 CALL FNDPDV : IS PROCESS IN SYSTEM?
665 003456 103004 BCC 10$ : BR IF YES
666 003460 ERRPT$ LOERR1,$EROUT : ELSE EVFNT LOGGING NOT SUPPORTED
667 003470 032767 000010 000000G 10$: BIT #OP$SIN,$OPTON : SINK NODE SPECIFIED?
668 003476 001004 BNE 20$ : BR IF YES
669 003500 052767 002000 000000G BIS #OP$KSK,$OPTON : ELSE DEFAULT IS KNOWN SINKS
670 003506 000426 BR 40$ : CONTINUE
671 003510 032767 002000 000000G 20$: BIT #OP$KSK,$OPTON : KNOWN SINKS SPECIFIED?
672 003516 001022 BNE 40$ : BR IF YES
673 003520 032767 000000G 000000G BIT #VF.SKA,$VFLAG : NODE ADDRESS SPECIFIED?
674 003526 001016 BNE 40$ : BR IF YES
675 003530 012701 000014G MOV #RQB+LO.SKN,R1 : POINT TO NODE NAME STRING
676 003534 005711 TST (R1) : EXECUTOR NODE SPECIFIED AS SINK NODE?
677 003536 001407 BEQ 30$ : BR IF YES
678 003540 CALL $FNNAME : FIND REMOTE NODE NAME BLOCK
679 003544 103004 BCC 30$ : BR IF FOUND IT
680 003546 ERRPT$ LOERR2,$EROUT : ELSE DISPLAY ERROR MESSAGE
681 003556 016767 000010G 000014G 30$: MOV #BFR+R.ADD,LO.SKA+RQB : STORE SINK NODE ADDRESS
682 003564 40$: CALL CKTYP1 : GET COMMAND TYPE STRING
683 003570 103004 BCC 50$ : BR IF SUCCESS
684 003572 ERRPT$ SYNERR,$EROUT : ELSE SYNTAX ERROR
685 003602 012703 001062' 50$: MOV #TEMP2,R3 : GET STORAGE FOR HEADER
686 003606 005767 000000G TST $EQUAL : SHOW SPECIFIC LOGGING TYPE?
687 003612 001003 BNE 60$ : BR IF NO
688 003614 012702 000313' MCV #LOSPEC,R2 : GET HEADER
689 003620 000413 BR 80$
690 003622 012705 001676' 60$: MOV #LOGTYP,R5 : POINT TO LOGGING SINK TYPE TABLE
691 003626 022767 000001 000000G CMP #OF$ACT,$EQUAL : SHOW ACTIVE LOGGING?
692 003634 001003 BNE 70$ : BR IF NO
693 003636 012702 000356' MCV #LOACTV,R2 : GET HEADER
694 003642 000402 BR 80$
695 003644 012702 000275' 70$: MOV #LCKNWN,R2 : MUST BE SHOW KNOWN LOGGING
696 003650 112223 80$: MOV (R2)+,(R3)+ : STORE THE HEADER
697 003652 001376 BNE 80$ :
698 003654 ERRPT$ LOMSG1,MSGOUT : DISPLAY THE HEADER
699
700 : DISPLAY LOGGING SINK TYPE
701

```

.TITLE VCEx - VNP LOAD CEX ROUTINES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - "SET CEX" ROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECnet-11M/S V3.0
DECnet-11M-PLUS V1.0
- 3.00 16-APR-82
DECnet-11M V3.1
DECnet-11M-PLUS V1.1
- 4.00 07-NOV-83
DECnet-11M V4.0
DECnet-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

VCEX - VNP LOAD CEX ROUTINES
CEPTR - INIT CETAB POINTERS

MACRO V05.03b Saturday 29-Jun-85 02:13 Page 11-1

```
611      BGT      10$
612
613      30$:     MOV      (SP)+,R1
614             MOV      (SP)+,R0
615             SWITCH1   ; RESET INPUT BUFFER
616             .ENDC
617
618             SWITCHC   ; RESET OUTPUT BUFFER
619             RETURN
620
```

VCEX

SYMBOL

SYMBOL

TSKSCCH

T.NAM

T.RCVL

T.ST2

T.TCBL

T2.FXD

VF.SS

VXBLK1

VXBLK2

VXBLK3

XBLK1

XBLK2

XBLK3

\$ALL15

\$AZFS

\$BFR

\$CCBNM

\$CEAVL

\$CET

\$CLFLG

\$CLSAV

\$CTBIO

\$CXOPT

\$DEA16

\$DECHM

\$DECPY

\$DFUIC

\$ERMSG

\$ERR44

\$ERR45

\$ERR46

\$ERR47

\$ERR48

\$ERR51

\$ERR53

\$ERR54

\$ERR55

\$ERR56

\$ERR57

\$ERR58

\$ERR59

\$ERR60

\$ERR61

\$ERR62

\$ERR63

\$ERR64

\$ERR65

\$ERR66

\$ERR67

\$ERR68

VCEX - VNP LOAD CEX ROUTINES
CEPTR - INIT CETAB POINTERS

MACRO V05.03b Saturday 29-Jun-85 02:13 Page 12

VCEX

VCEX CPEATED BY MACRO ON 27-JUN-85 AT 02:14 PAGE 3 C 3

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
TSKSCH	001120 R	#12-635 14-738
T.NAM	000006	12-644 12-646
T.RCVL	000012	14-787 14-789 *14-803 *14-804
T.ST2	000034	14-751
T.TCBL	000030	*12-638 12-640
T2.FXD	= 002000	14-751
VF.SS	= ***** GX	14-749
VXBLK1	000552 R	#6-193
VXBLK2	000564 R	6-193 #6-196
VXBLK3	000576 R	6-196 #6-199
XBLK1	000514 R	6-171 #6-180 9-495 10-531
XBLK2	000526 R	6-180 #6-183
XBLK3	000540 R	6-183 #6-186
\$ALL15	= ***** GX	13-670 14-779
\$AZFS	= ***** GX	7-309 8-354 9-421 9-446 9-453
\$BFR	= ***** GX	7-319 7-326 7-338 7-339 *8-349 *8-362 8-381 11-618 12-637
		13-677 13-713 14-751 14-787 14-789 14-792 *14-799 *14-803 *14-804
		*14-807 *14-811 *14-812 *14-813 *14-814 *14-815 *14-816 14-848
\$CCBNM	= ***** GX	11-581
\$CEAVL	= ***** GX	*7-338 *7-339 *7-340 8-362
\$CET	= ***** GX	8-352 9-451
\$CLFLG	= ***** GX	*9-409
\$CLSAV	= ***** GX	9-408
\$CTBIO	= ***** GX	8-355
\$CXOPT	= ***** GX	*11-570 11-583 11-584
\$DEA16	= ***** GX	13-720 14-801
\$DECHM	= ***** GX	8-374
\$DECP7	= ***** GX	8-379
\$DFUIC	= ***** GX	7-300 8-351 9-412 9-435 9-450
\$ERMSG	= ***** GX	15-866
\$ERR44	= ***** GX	6-235
\$ERR45	= ***** GX	6-236
\$ERR46	= ***** GX	6-237
\$ERR47	= ***** GX	6-238
\$ERR48	= ***** GX	9-506
\$ERR51	= ***** GX	6-243
\$ERR53	= ***** GX	6-245
\$ERR54	= ***** GX	6-246
\$ERR55	= ***** GX	6-247
\$ERR56	= ***** GX	6-249
\$ERR57	= ***** GX	6-250
\$ERR58	= ***** GX	6-251
\$ERR59	= ***** GX	6-255
\$ERR60	= ***** GX	6-256
\$ERR61	= ***** GX	6-257
\$ERR62	= ***** GX	6-258
\$ERR63	= ***** GX	6-259
\$ERR64	= ***** GX	6-261
\$ERR65	= ***** GX	6-262
\$ERR66	= ***** GX	6-263
\$ERR67	= ***** GX	6-267
\$ERR68	= ***** GX	6-268

VCFG -
\$NLCFG

304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360

```

304 000202          CALL    $QPVC          ; IS IT PVC$DF?
305 000206 103753    BCS     50$           ; BR IF NO
306 000210 000167    JMP     190$         ; ELSE FINISH UP
307
308 000214 032767    000000G 000000G 80$: BIT    #MS.LLC,$MISS ; ARE LLC PARAMETERS MISSING?
309 000222 001124    BNE     170$         ; IF NE, YES
310
311          ; ARE THE DDM/DLC PROCESS EXTENDS MISSING ?
312
313 000224 032767    000000C 000000G      BIT    #<MS.INC!MS.EXT>,$MISS ; ARE THE PROCESS EXTENDS MISSING ?
314 000232 001111    BNE     160$         ; IF NE, YES
315          ; NOTE - THE PROCESS EXTENDS ARE SET FOR PROCESS
316          ; ONLY. LLC'S ARE HANDLED SEPARATELY
317
318 000234          90$: CALL    READ          ; READ NEXT RECORD
319 000240 103437    BCS     123$         ; IF CS, EOF
320 000242          CALL    QSLT          ; IS IT SLT$DF?
321 000246 103040    BCC     130$         ; IF CC, YES
322 000250          CALL    QDDM          ; IS IT DDM$DF?
323 000254 103767    BCS     90$          ; IF CS, NO
324
325          ; DDM$DF SEEN
326
327
328 000256          100$: CALL    READ          ; READ NEXT RECORD
329 000262 103575    BCS     121$         ; IF CS, EOF
330 000264          CALL    QCNT          ; IS IT CNT$DF?
331 000270 103772    BCS     100$         ; IF CS, NO - KEEP LOOKING
332 000272 005767    000000G      TST     $MISS ; ANY PARAMETERS STILL MISSING?
333 000276 001454    BEQ     155$         ; IF EQ, NO
334 000300          110$: CALL    READ          ; READ NEXT RECORD
335 000304 103567    BCS     131$         ; IF CS, EOF
336 000306          CALL    QUNT          ; IS IT UNT$DF?
337 000312 103553    BCS     101$         ; IF CS, NO - ERROR
338 000314 026767    000000G 000152'    CMP     $MUX,UCNT ; HAVE ALL UNT$DF'S BEEN SEEN?
339 000322 001366    BNE     110$         ; IF NE, NO
340 000324 042767    000000G 000000G      BIC     #MS.MUX,$MISS ; INDICATE ALL MUX PARAMETERS SEEN
341 000332 001436    BEQ     155$         ; IF EQ, ALL DONE
342 000334          120$: CALL    READ          ; READ NEXT RECORD
343 000340 103543    123$: BCS     111$         ; IF CS, EOF
344 000342          CALL    QSLT          ; IS IT SLT$DF?
345 000346 103772    BCS     120$         ; IF CS, NO
346
347          ; SLT$DF SEEN
348
349
350 000350          130$: CALL    READ          ; GET NEXT RECORD
351 000354 103546    BCS     141$         ; IF CS, EOF
352 000356          CALL    QSTA          ; IS IT STAS$DF?
353 000362 103372    BCC     130$         ; IF CC, YES - LOOK FOR MORE OF THEM
354 000364          CALL    QPLT          ; IS IT PLT$DF?
355 000370 103002    BCC     135$         ; IF YES - BRANCH
356 000372          CALL    QADD          ; IS IT ADD$DF?
357 000376 042767    000000G 000000G 135$: BIC     #MS.MTP,$MISS ; INDICATE ALL MTP PARAMETERS SEEN
358 000404 001411    BEQ     155$         ; IF EQ, ALL DONE
359 000406          140$: CALL    QDDM          ; IS IT DDM$DF?
360 000412 103321    BCC     100$         ; IF CC, YES

```

VCFG - V
SLT\$DF S
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790


```
770 001512      TRANS  <','>
771 001512      STATES
772 001512      TRANS  $NUMBER,$EXIT,$PRI
773
774
775
776
777      ; FLAG BIT DEFINITIONS PLUS TRAILING COMMA
778      ;
778 001512      STATES  BITS
779 001512      TRANS  %LF.X2P%,BITS1,,FG.X2P,CFGFLG
780 001512      TRANS  %ZF.X3P%,BITS1,,FG.X3P,CFGFLG
781 001512      TRANS  $RAD50,BITS1
782 001512      TRANS  <','>,$EXIT
783 001512      STATES  BITS1
784 001512      TRANS  < '!> ,BITS2
785 001512      TRANS  '+,BITS2
786 001512      TRANS  <','>,$EXIT
787 001512      STATES  BITS2
788 001512      TRANS  %LF.X2P%,BITS1,,FG.X2P,CFGFLG
789 001512      TRANS  %ZF.X3P%,BITS1,,FG.X3P,CFGFLG
790 001512      TRANS  $RAD50,BITS1
```

1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335

```

1277
1278          : INTERRUPT VECTOR (CNT$DF)
1279
1280 002432 032767 000000G 000000G C.VECT: BIT      #MS.VCT,$MISS ; IS VECTOR MISSING?
1281 002440 001421          BEQ      10$      ; IF EQ, NO
1282 002442 005767          TST      ,PNUMH   ; MAKE SURE IT'S A VALID VECTOR
1283 002446 001017          BNE      101$     ;
1284 002450 032767 000003 000000G      BIT      #3, PNUMB   ;
1285 002456 001013          BNE      101$     ;
1286 002460 026767 000000G 000000G      CMP      ,PNUMB,$MXVCT ; AND THAT IT'S IN OUR SYSTEM
1287 002466 103012          BHS      111$     ;
1288 002470 016767 000000G 000000G      MOV      ,PNUMB,$$VECT ; STORE VECTOR VALUE
1289 002476 042767 000000G 000000G      BIC      #MS.VCT,$MISS ; IT'S NO LONGER MISSING
1290 002504          10$: RETURN
1291 002506          101$: MSG$R 1C      ; ILLEGAL VECTOR VALUE
1292 002514          111$: MSG$R 1D      ; VECTOR NOT IN SYSTEM
1293
1294          :
1295          : CSR (CNT$DF)
1296
1297 002522 032767 000000G 000000G C.CSR: BIT      #MS.CSR,$MISS ; IS CSR MISSING?
1298 002530 001410          BEQ      10$      ; IF EQ, NO
1299 002532          CALL      GETCSR      ; GET THE CSR ADDRESS
1300 002536          103406          BCS      101$     ;
1301 002540          010067 000000G      MOV      RO,$$CSR      ; STORE CSR VALUE
1302 002544 042767 000000G 000000G      BIC      #MS.CSR,$MISS ; IT'S NO LONGER MISSING
1303 002552          10$: RETURN
1304 002554          101$: MSG$R 1E      ; ILLEGAL CSR VALUE
1305
1306          :
1307          : RETURN NUMERIC CSR VALUE IN RO
1308
1309 002562 005767 000000G      GETCSR: TST      ,PNUMH   ; MAKE SURE IT'S A VALID CSR
1310 002566 001404          BEQ      10$      ;
1311 002570 026727 000000G 000003      CMP      ,PNUMH,#3      ; DOES THE EXCESS = 600000 ?
1312 002576 001011          BNE      20$      ; IF NE, NO
1313 002600 016700 000000G      10$: MOV      ,PNUMB,RO      ; GET THE LOW ORDER NUMBER
1314 002604 032700 000003          BIT      #3,RO      ; IS IT AN EVEN MULTIPLE OF 4 ?
1315 002610 001004          BNE      20$      ; IF NE, NO
1316 002612 026727 000000G 160000      CMP      ,PNUMB,#160000 ; IS IT IN THE I/O PAGE ?
1317 002620 101001          BHI      30$      ; IF HIS, YES
1318 002622 000261          20$: SEC      ; ILLEGAL VALUE
1319 002624          30$: RETURN
1320
1321          :
1322          : INTERRUPT PRIORITY (CNT$DF)
1323
1324 002626 032767 000000G 000000G C.PRI: BIT      #MS.PRI,$MISS ; IS PRIORITY MISSING?
1325 002634 001424          BEQ      10$      ; IF EQ, NO
1326 002636 005767 000000G      TST      ,PNUMH   ; MAKE SURE IT'S A VALID PRIORITY
1327 002642 001022          BNE      101$     ;
1328 002644 026727 000000G 000007      CMP      ,PNUMB,#7      ;
1329 002652 101016          BHI      101$     ;
1330 002654 016700 000000G      MOV      ,PNUMB,RO      ; GET PRIORITY VALUE
1331 002660 001413          BEQ      101$     ; IF ZERO, ILLEGAL
1332 002662          ASL$      5,RO      ; CONVERT TO PSW PRIORITY VALUE
1333 002674 010067 000000G      MOV      RO,$$PRI      ; STORE IT

```

VCFG -
Symbol
Size of
Size of
Operati
Elapsed
DB2:VCF

VCFG - VNP SCAN CONFIGURATION F MACRO V05.03b Monday 15-Jul-85 19:04^{C 7} Page 35-3
Symbol table

Size of work file: 18149 Words (71 Pages)
Size of core pool: 17608 Words (67 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:03:21.58
DB2:VCFG.T34,[132,134]VCFG/CR/-SP=DB2:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VCFG

VCFP

575 001030

20\$: RETURN

VCFP

SYMBOL

SYMBOL

LN. 75

L. COS

L. CTL

L. CVA

L. DDN

L. DDS

L. DCC

L. DLM

L. DLS

L. FLC

L. KRE

L. LEN

L. MPP

L. NMS

L. NS

L. OWA

L. UN

MAXI

MOVE

MRKFI

MXAC

MXAC

MXAC

MXNO

NEXT

N\$SV

OPEN

PCKB

PF. P

PF. X

PF. X

PORT

PVST

PV. O

PV. O

RBRA

READ

RNAC

RNAL

RNET

RNWL

RNNO

RNPA

RNPL

RNTL

RNUL

RNJS

RNWE

RTSP

R\$SE

R\$S1

VCFP CREATED BY MACRO ON 29-JUN-85 AT 02:18 PAGE 3 C 10
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
LN.TRI	= 000006	#5-67
L.COST	000015	#5-67
L.CTL	000012	#5-67
L.CVA	177776	#5-67
L.DDM	000002	#5-67
L.DDS	000004	#5-67
L.DLC	000003	#5-67
L.DLM	000006	#5-67
L.DLS	000010	#5-67
L.FLG	000000	#5-67
L.IRBA	000016	#5-67
L.LEN	= 000022	#5-67
L.MPF	000022	#5-67
L.NMST	000020	#5-67
L.NSTA	000014	#5-67
L.OWNR	000021	#5-67
L.UNT	000013	#5-67
MAXID	= 000020	#6-96 7-184 7-186 7-226 7-243 19-733 19-745
MOVE	001430 RG	19-753 #20-775
MRKFL	000344 RG	#7-143
MXACAC	= 000020	#6-100 7-251
MXACPA	= 000010	#6-99 7-249
MXACUS	= 000020	#6-98 7-247
MXNOD	= 000006	#6-97 7-245
NEXT	000352 RG	#7-148 18-693 *18-694
N\$SVCT	= *****	16-632
OPEN	000400 R	9-307 #10-419
PCKBCD	001032 RG	#15-593
PF.PSV	= ***** GX	9-363 9-381
PF.XDF	= ***** GX	19-737
PF.XGA	= ***** GX	9-312 9-326 9-344
PORTNO	000354 RG	#7-149
PVSTA	000402 RG	#7-174
PV.OFF	= 000001 G	#7-175
PV.ON	= 000002 G	#7-176
RBRA	= 000076	#6-87
READ	000442 R	9-308 9-314 9-321 9-348 9-359 9-373 9-377 9-383 #11-444
RNACCT	000745 RG	#7-251
RNAL	000744 RG	#7-250
RNETWK	000663 RG	#7-243
RNWL	000703 RG	#7-244
RNNODE	000704 RG	#7-245
RNPASS	000734 RG	#7-249
RNPL	000733 RG	7-206 #7-248
RNTL	000662 RG	#7-242
RNUL	000712 RG	7-205 #7-246
RNUSER	000713 RG	#7-247
RNWEND	= ***** GX	9-352
RTSPC	000630 RG	8-259 8-260 8-261 8-262 8-263 8-264 8-265 8-266 8-267
R\$EIS	= *****	15-604
R\$IID	= *****	15-604

VCIO -
 Symbol
 \$PUTRC
 . ABS.
 DATA
 . BUF
 Errors
 *** As
 Work
 Work
 Size o
 Size o
 Operat
 Elaps
 SY:VCI

VC10 - LOAD CETAB IMAGE MACRO V05.03b Saturday 29-Jun-85 02:18 Page 10-2
Symbol table

\$PUTRC= ***** GX \$RADDR= ***** GX \$READ = ***** GX \$RLBL = ***** GX \$SAVVR= ***** GX

. ABS. 001000 000 (RW,I,GBL,ABS,OVR)
000542 001 (RW,I,LCL,REL,CON)
DATA 000002 002 (RW,D,LCL,REL,CON)
.BUF 000002 003 (RW,D,LCL,REL,CON)
Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 11930 Words (47 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:11.30

SY:VC10.V2,[132,134]VC10/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VC10

VC10R
SYMB
SYMB
AUXPD
CLORE
C.ARS
C.LGT
C.ROT
C.SYS
C.TCB
FNDPD
F.RSI
LD.AL
LD.CE
LD.CI
LD.LI
LD.LO
LD.MO
LD.NO
LD.OB
LD.PR
LD.AD
LD.AD
LD.CL
LD.CO
LD.CO
LD.CO
LD.CO
LD.CT
LD.DE
LD.DT
LD.EV
LD.GR
LD.HC
LD.HT
LD.IN
LD.IN
LD.LD
LD.LI
LD.LI
LD.LI
LD.MA
LD.MC
LD.NA
LD.NE
LD.NM
LD.OF
LD.ON
LD.OL
LD.OW
LD.PA
LD.PA
LD.PP
LD.RE
LD.RP
LD.SC

VCLORM CREATED BY MACRO ON 29-JUN-85 AT 02:18 PAGE 1 C 12

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
AUXFDV	000000 R	#6-67 *7-87 *7-91 7-118
CLQREM	000002 RG	#7-86
C.ARS	000014	7-120
C.LGTH	= 000020	7-101 7-133
C.RQT	000002	7-108
C.SYST	= 000006	7-108
C.TCB	000004	7-104 7-125
FNDPDV	000242 R	7-89 #8-159
F.RSIZ	= ***** GX	*7-94 *7-101
LD.ALI	= 000030	#5-59
LD.CEX	= 000024	#5-59
LD.CIR	= 000035	#5-59
LD.LIN	= 000025	#5-59
LD.LOG	= 000031	#5-59
LD.MOD	= 000037	#5-59
LD.NOD	= 000027	#5-59
LD.OBJ	= 000032	#5-59
LD.PRO	= 000026	#5-59
LO.ADD	000022	#5-59
LO.ADR	000052	#5-59
LO.CLS	000002	#5-59
LO.CON	000006	#5-59
LO.COP	000010	#5-59
LO.COS	000020	#5-59
LO.CSR	000010	#5-59 5-59
LO.CTL	000032	#5-59 5-59
LO.DES	000002	#5-59
LO.DTE	000047	#5-59
LO.EVT	000004	#5-59
LO.GRO	000024	#5-59
LO.HOS	000040	#5-59
LO.HTM	000032	#5-59
LO.INA	000062	#5-59
LO.INC	000060	#5-59
LO.LDT	000046	#5-59
LO.LID	000030	#5-59
LO.LIN	000004	#5-59
LO.LTM	000034	#5-59
LO.MAC	000022	#5-59
LO.MDE	000024	#5-59
LO.NAM	000044	#5-59
LO.NET	= 000002	#5-59
LO.NNM	000022	#5-59
LO.OFL	000012	#5-59
LO.ONM	000004	#5-59
LO.OUT	000056	#5-59
LO.OWN	000026	#5-59
LO.PAR	000010	#5-59
LO.PRI	000012	#5-59
LO.RET	000064	#5-59
LO.RPA	000030	#5-59
LO.SCR	000016	#5-59

C 13

```

245          BNE      14$          ; IF NOT - BRANCH          ;[MP01]
246          BICB     #US.OFL,U.ST2+DEVBUF ; SET DEVICE ONLINE ;[MP01]
247          14$:                                     ;[MP01]
248          .ENDC                                     ;[MP01]
249
250 000166          PUTRC   R0,R1          ; REWRITE UCB
251 000202 060100   ADD     R1,R0          ; POINT AT NEXT UCB
252 000204 005367 000172' DEC     UNITS          ; ONE LESS UCB
253 000210 003355          BGT     15$          ; IF GT, MORE UCB'S
254 000212          20$:          SWITCHO          ; DEFAULT TO $BFR
255 000220          SWITCHI          ; DEFAULT TO $BFR
256 000226          RETURN
257
258          ;
259          ; ERROR CONDITION
260          ;
261 000230 012701 000146' 101$: MOV     #DUMMY-4,R1          ; POINT AT DUMMY SYMBOL
262 000234          MSG$R  2Y          ; DCB DISPATCH TABLE ADDRESS UNDEFINED
  
```

D 13

```

741                                     .SBTTL REP9 - REPLACE DEVICE NAME
742                                     ;
743                                     ; REP9 - REPLACE DEVICE NAME
744                                     ;
745                                     ; INPUTS:
746                                     ;     NONE
747                                     ;
748                                     ; OUTPUTS:
749                                     ;     NONE
750                                     ;
751 002346 112314 REP9:  MOVB  (R3)+,(R4)      ; COPY THE FORMAT STRING
752 002350 122724 000052  CMPB  #'*,(R4)+    ; LOOKING FOR THE FIRST ASTERISK
753 002354 001374      BNE   REP9            ;
754 002356 116764 000000G 177777  MOVB  $$DEV,-1(R4)    ; STORE DEVICE NAME
755 002364 116724 000001G      MOVB  $$DEV+1,(R4)+ ;
756 002370      RETURN                      ;
757
758 000001                                     .END

```

```

162 000246      103      154      145 PCLEAR: .ASCIZ /Cleared/
163
164           :
165           : ASCIZ STRINGS FOR LOGGING HEADER
166           :
167 000256      101      143      164 LOACTV: .ASCIZ /Active logging/
168 000275      113      156      157 LOKNWN: .ASCIZ /Known logging/
169 000313      114      157      147 LOSPEC: .ASCIZ /Logging/
170
171           :
172           : ASCIZ STRINGS FOR LOGGING SINK TYPE
173           :
174 000323      115      157      156 LOMON: .ASCIZ /Monitor/
175 000333      106      151      154 LOFIL: .ASCIZ /File/
176 000340      103      157      156 LOCON: .ASCIZ /Console/
177
178 000350      116      157      040 LONOEV: .ASCIZ /No events/
179
180           :
181           : ASCIZ STRINGS FOR NODE HEADER
182           :
182 000362      113      156      157 NKKNOWN: .ASCIZ /known nodes/
183 000376      114      157      157 NLOOP: .ASCIZ /Loop nodes/
184 000411      116      157      144 NSPEC: .ASCIZ /Node/
185
186           :
187           : ASCIZ STRINGS FOR ROUTING TYPE FOR EXECUTOR NODE
188           :
189 000416      122      157      165 ROUTNG: .ASCIZ /Routing/
190 000426      116      157      156 RROUT: .ASCIZ /Nonrouting/
191
192           :
193           : ASCIZ STRINGS FOR LINE HEADER
194           :
195 000441      101      143      164 LACTIV: .ASCIZ /Active lines/
196 000456      113      156      157 LKNOWN: .ASCIZ /Known lines/
197 000472      114      151      156 LSPEC: .ASCIZ /Line/
198
199           :
200           : ASCIZ STRING FOR CIRCUIT SERVICE DISPLAY
201           :
202 000477      105      156      141 ENABLE: .ASCIZ /Enabled/
203 000507      104      151      163 DISABL: .ASCIZ /Disabled/
204
205           :
206           : ASCIZ STRINGS FOR CIRCUIT HEADER
207           :
208 000520      101      143      164 CACTIV: .ASCIZ /Active circuits/
209 000540      113      156      157 CKNOWN: .ASCIZ /Known circuits/
210 000557      103      151      162 CSPEC: .ASCIZ /Circuit/
211
212           :
213           : ASCIZ STRINGS FOR LINE TYPE
214           :
214 000567      104      104      103 LCNTRL: .ASCIZ /DDCMP Control/
215 000605      104      104      103 LPOINT: .ASCIZ /DDCMP Point/
216 000621      104      104      103 LTRIB: .ASCIZ /DDCMP Tributary/
217 000641      130      062      065 LX25: .ASCIZ /X25/
218 000645      114      101      120 LLAB: .ASCIZ /LAPB/

```

VDIS - VNP SHOW COMMANDS
 - SHOW LOGGING

MACRO V05.03b Monday 15-Jul-85 12:14 Page 12-1

```

72 003664 005767 000000G      IST    $QUAL      ; SHOW SPECIFIC LOGGING?
73 003670 001407              BEQ    100$      ; BR IF YES
704 003672 042767 000700 000000G 90$: BIC    #<OP$CON!OP$FIL!OP$MON>,$OPTON ; CLEAR SINK TYPE BITS
705 003700 012501              MOV    (R5)+,R1   ; GET BIT TO SET
706 003702 001455              BEQ    150$      ; BR IF END OF TABLE
707 003704 050167 000000G      BIS    R1,$OPTON ; SET BIT FOR NEXT SINK TYPE
708 003710 012767 177776 175264 100$: MOV    #-2,TEMP3 ; INDICATE NO SINK NODE DISPLAYED
709 003716              CALL    DLOTYPE ; DISPLAY LOGGING SINK TYPE
710 003722 103742              BCS    140$      ; BR IF FINISHED
711              ;
712              ; DISPLAY LOGGING STATE
713              ;
714 003724              CALL    DLOSTA          ; DISPLAY LOGGING STATE
715              ;
716              ; DISPLAY LOGGING SINK NAME
717              ;
718 003730              CALL    DLONAM          ; DISPLAY LOGGING SINK NAME
719              ;
720              ; GET NEXT EVENT TO PROCESS
721              ;
722 003734 010546              MOV    R5,-(SP)    ; SAVE R5
723 003736 012705 000016      MOV    #F.LEN,R5   ; GET LENGTH OF FILTER CONTROL BLOCK
724 003742 012703 000000C      MOV    #F.FILHD,R3 ; POINT TO FILTER BLOCK LISTHEAD
725 003746              CALL    NXTBLK        ; GET NEXT FILTER BLOCK IN LIST
726 003752 103425              BCS    130$      ; BR IF FINISHED
727 003754 016767 000002G 000000G      MOV    F.CLS+$BFR,$RSIZE ; GET LENGTH OF BLOCK
728 003762 042767 177700 000000G      BIC    #^C<FF,MSK>,$RSIZE ;
729 003770 022767 000016 000000G      CMP    #F.LEN,$RSIZE ;
730 003776 001402              BEQ    120$      ; IS THIS THE CORRECT SIZE?
731 004000              GETAD              ; BR IF YES
732 004004 030463 000014      BIT     R4,F.FLG(R3) ; READ IN ENTIRE FILTER BLOCK
733 004010 001756              BEQ    110$      ; IS THIS THE CORRECT SINK TYPE?
734              ;
735              ; CHECK FOR MATCH ON SINK NODE AND DISPLAY IT IF NECESSARY
736              ;
737 004012              CALL    CHKSNK        ; CHECK FOR MATCH ON SINK TYPE
738 004016 103753              BCS    110$      ; BR IF NO MATCH
739              ;
740              ; DISPLAY EVENTS
741              ;
742 004020              CALL    DLOEVT        ; DISPLAY EVENT MESSAGE
743 004024 000750              BR     110$      ; GET NEXT FILTER BLOCK
744              ;
745              ; CHECK FOR END
746              ;
747 004026 012605 130$: MOV    (SP)+,R5
748 004030 005767 000000G      140$: IST    $QUAL      ; SHOW ACTIVE OR KNOWN LOGGING?
749 004034 001316              BNE    90$      ; BR IF YES
750 004036              150$: ERRPT$ LOMSG5,MSGOUT ; DISPLAY BLANK LINE
751              ;
752 004046              DIR$ #TIDET          ; DETACH TERMINAL
753 004054              RETURN
754              ;
755              ; ENABL LC
756              ;
757              ;
758              ; DISPLAY MESSAGES

```

54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72

```
.SBTTL  MACRO DEFINITIONS
;***
; LIBRARY MACROS
;***
.MCALL  ASL$,EMSG$,GETRV,PUTRC,GETAD,PUTAD,PCBDF$,SWTCHO,NTLDF$
.MCALL  OPTDF$,DHBDF$,ERMSG$,ERBLK$,ERRPT$,CALLR,CLKDF$,TCBDF$
.MCALL  SAVRG,RESRG,SWTCHI,PKTDF$

.ENABL  LC

PCBDF$  : DEFINE PCB OFFSETS
NTLDF$  : DEFINE NTL OFFSETS
OPTDF$  : COMM/EXEC OPTIONS
DHBDF$  : DECNET HOME BLOCK OFFSETS
CLKDF$  : DEFINE CLOCK QUEUE OFFSETS
TCBDF$  : DEFINE TCB OFFSETS
PKTDF$  : DEFINE PACKET OFFSETS
```

```

622
623
624
625
626
627
628
629
630
631
632
633
634
635 001120
636 001120
637 001140 012705 000000G
638 001144 011565 000030
639
640 001150 016567 000030 000000G 10$:
641 001156 000261
642 001160 001411
643 001162
644 001166 021065 000006
645 001172 001366
646 001174 026065 000002 000010
647 001202 001362
648 001204

      **--TSKSCH-SEARCH TCB'S FOR SPECIFIED TASK
      :+
      :
      : INPUTS:
      :   RO      - POINTER TO RAD50 TASK NAME
      :
      : OUTPUTS:
      :   $RADDR  - ADDRESS OF TCB
      :   $BFR    - IMAGE OF TCB
      :   C-BIT SUCCESS/FAILURE
      :   R3,R4,R5 DESTROYED
      :
      :
      : TSKSCH:
      :   GETRV   .TSKHD,.TLGTH ; READ IN LISTHEAD
      :   MOV     #BFR,R5      ; POINT AT BUFFER BASE
      :   MOV     (R5),T.TCBL(R5) ; INITIALIZE FOR LOOP
      :
      :   MOV     T.TCBL(R5),$RADDR ; SET TCB ADDRESS
      :   SEC                     ; ASSUME ERROR
      :   BEQ     30$             ; NULL TASK ( END OF LIST )
      :   GETRV   (R0),T.NAM(R5) ; READ IN TCB
      :                       ; MUST MATCH BOTH
      :   CMP     10$
      :   BNE     2(R0),T.NAM+2(R5) ; HALVES OF NAME
      :   CMP     10$
      :   BNE     10$
      :   C-BIT IS CLEAR
      :
      :   30$: RETURN

```

VCEX
SYMBOL
SYMBOL
\$ERR9E
\$FMASK
\$GETRV
\$LLCTA
\$NTLH8
\$NTLPT
\$PDVNM
\$PDVTA
\$PRCLN
\$PREXT
\$PRINC
\$PRIO
\$PRLEN
\$PRLLN
\$PUTRC
\$RADDR
\$SAVAL
\$SCEX
\$SLTMA
\$SLTNM
\$SYMBL
\$TIOUT
\$VFLAG
\$VIMI
\$V

\$MUMX
\$\$\$

\$\$\$
\$.ABTIM
\$.CEAVL
\$.CEPWR
\$.CLINS
\$.CLKHD
\$.CXALL
\$.CXCSR
\$.CXLBR
\$.CXLB1
\$.CXLB2
\$.CXLEN
\$.CXPCB
\$.CXSVM
\$.CXUNL
\$.CXVEC
\$.CXVSM
\$.FRK4D

VC EX CREATED BY MACRO ON 29-JUN-85 AT 02:14 PAGE 4 D 3

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
\$ERR9E	= ***** GX	6-269
\$FMASK	= ***** GX	*8-392
\$GETRV	= ***** GX	*8-348 7-325 8-379 12-636 12-643 13-685 13-692 13-697 13-708
\$LLCTA	= ***** GX	14-786 11-567 11-574
\$NTLHB	= ***** GX	*10-523 *10-528 10-532 10-547
\$NTLPT	= ***** GX	*8-368
\$PDVNM	= ***** GX	*11-569
\$PDVTA	= ***** GX	*11-565 *11-574 11-580 11-581 11-583
\$PRCLN	000374 RG	#6-109
\$PREXT	000374 RG	#6-110
\$PRINC	000374 RG	#6-111
\$PRIO	= ***** GX	7-313
\$PRLEN	000374 RG	#6-112
\$PRLLN	000374 RG	#6-113
\$PUTRC	= ***** GX	7-337 8-350 8-363 8-385 11-581 11-585 13-710 13-712 14-800
\$RADDR	= ***** GX	14-805 14-837 *7-318 *7-325 *7-333 *8-350 *8-363 *8-379 *11-581 *11-585 *12-676
\$SAVAL	= ***** GX	*12-640 *13-685 *13-692 *13-697 *13-708 *13-712 *14-786 *14-789
\$SCEX	000000 RG	*14-837
\$SLTMA	= ***** GX	15-864 *11-575
\$SLTNM	= ***** GX	*11-566
\$SYMBL	= ***** GX	*11-568
\$TIOUT	= ***** GX	9-430 9-448 9-455
\$VFLAG	= ***** GX	15-867
\$VIMI	= ***** GX	14-749
\$S	= 177777	8-366
\$SMUX	= ***** GX	6-93 #6-93 #6-94 6-95 6-95 #6-95 #6-96 6-97
\$S\$	= 000226 R	6-97 #6-97 #6-98 6-99 6-99 #6-99 #6-100 6-101 6-101
\$S\$S	= 000252 R	#6-101 #6-102 #6-102 #6-94 #6-95 6-95 #6-97 6-98 #6-99 6-100 #6-101
.ABTIM	= ***** GX	8-368
.CEAVL	= ***** GX	#6-93 6-94 #6-95 6-95 #6-97 6-98 #6-99 6-100 #6-101
.CEPWR	= ***** GX	6-102
.CLINS	= ***** GX	#6-94 #6-96 #6-98 #6-100 #6-102
.CLKHD	= ***** GX	13-685
.CXALL	000062	7-317 8-363
.CXCSR	000004	8-349
.CXLBR	000060	6-164
.CXLB1	000030	13-691
.CXLB2	000044	*10-550
.CXLEN	000064	*10-523
.CXPCB	000006	*10-551
.CXSYM	000010	*10-528
.CXUNL	000024	10-532
.CXVEC	000026	#5-65
.CXVSM	000016	#5-65
.FRK4D	= ***** GX	6-167

VC FG -
\$NL CFG

361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500

```

361 000414          CALL      READ      ; READ NEXT RECORD
362 000420          BCC      140$      ; IF CC, NOT EOF
363 000422          150$: EMSG$R 1Y      ; DDM$DF MISSING (EOF)
364
365 000430 032767 000001 000136' 155$: BIT      #FG.X2P,CFGFLG ; DO WE NEED X2P$DF?
366 000436 001475          BEQ      190$      ; BR IF NO
367 000440          157$: CALL      READ      ; READ NEXT RECORD
368 000444 103523          BCS      171$      ; BR IF EOF
369 000446          CALL      $QX2P      ; LOOK FOR X2P$DF
370 000452 103772          BCS      157$      ; BR IF NOT FOUND
371 000454 000466          BR      190$      ; CONTINUE
372
373          ; LOOK FOR DDM/DLC PROCESS EXTENDS
374
375 000456          160$: CALL      READ      ; READ NEXT RECORD
376 000462 103757          BCS      150$      ; IF CS, DDM/DLC MISSING
377 000464          CALL      QDDM      ; IS IT A DDM OR DLC ?
378 000470 103772          BCS      160$      ; IF CS, NO .. KEEP LOOKING
379 000472 000457          BR      190$      ; ELSE, LEAVE
380
381          ; LOOK FOR LLC$DF
382
383 000474          170$: CALL      READ      ; READ NEXT RECORD
384 000500 103477          BCS      151$      ; IF CS, EOF
385 000502          CALL      QLLC      ; IS IT LLC$DF?
386 000506 103772          BCS      170$      ; IF CS, NO
387
388          ; LOOK FOR FEA$DF
389
390 000510 005067 000000G          CLR      $FEATR      ; ASSUME NO FEATURES WORD
391 000514          180$: CALL      READ      ; READ NEXT RECORD
392 000520 103434          BCS      188$      ; IF CS, EOF
393 000522          CALL      QFEA      ; IS IT FEA$DF ?
394 000526 103016          BCC      185$      ; BR IF YES
395 000530 032767 000002 000136' 185$: BIT      #FG.X3P,CFGFLG ; IS X3P$DF NEEDED?
396 000536 001766          BEQ      180$      ; BR IF NO
397 000540          CALL      $QX3P      ; IS IT X3P$DF?
398 000544 103763          BCS      180$      ; BR IF NO
399
400          ; X3P$DF FOUND
401
402 000546          182$: CALL      READ      ; READ NEXT RECORD
403 000552 103427          BCS      190$      ; BR IF END OF FILE
404 000554          CALL      QFEA      ; IS IT FEA$DF?
405 000560 103772          BCS      182$      ; BR IF NO
406 000562 000423          BR      190$      ; ELSE EXIT
407
408          ; FEA$DF FOUND
409
410 000564 032767 000002 000136' 185$: BIT      #FG.X3P,CFGFLG ; IS X3P$DF NEEDED?
411 000572 001417          BEQ      190$      ; BR IF NO
412 000574          187$: CALL      READ      ; READ NEXT RECORD
413 000600 103442          BCS      161$      ; BR IF EOF
414 000602          CALL      $QX3P      ; IS IT X3P$DF?
415 000606 103772          BCS      187$      ; BR IF NO
416 000610 000410          BR      190$      ; EXIT
417

```

792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833


```

792 .SBTTL STAD$F STATE TABLE
793 ;
794 ; STATION TABLE DEFINITION (STAD$F)
795 ;
796 STATES STAD$F
797 TRANS %STAD$F%,,,1,SYNERR
798
799 STATES STAD$F ; STATION NUMBER
800 TRANS $NUMBR,,ST.NUM
801
802 STATES STAD$F
803 TRANS <','>
804
805 STATES STAD$F ; PROCESS NAME
806 TRANS $RAD50
807
808 STATES STAD$F
809 TRANS <','>,STADF1
810 TRANS $LAMDA,$EXIT ; THE REST IS OPTIONAL
811
812 STATES STAD$F
813 TRANS !BITS,STADF2 ; FLAG BITS
814 TRANS $LAMDA,$EXIT
815
816 STATES STAD$F
817 TRANS STADF2 ; STATION COST
818 TRANS $NUMBR,,ST.CST
819
820 STATES STAD$F
821 TRANS !END,$EXIT
822 TRANS <','>
823
824 STATES STAD$F
825 TRANS $NUMBR,,ST.APR ; ACTIVE POLLING RATIO
826
827 STATES STAD$F
828 TRANS !END,$EXIT
829 TRANS <','>
830
831 STATES STAD$F
832 TRANS $NUMBR,END,ST.HTM ; HELLO TIMER, LISTEN TIMER/2
833

```

1334
 1335
 1336
 1337
 1338
 1339
 1340
 1341
 1342
 1343
 1344
 1345
 1346
 1347
 1348
 1349
 1350

```

1334 002700 042767 000000G 000000G      BIC      #MS.PRI,$MISS      ; IT'S NO LONGER MISSING
1335 002706      10$:      RETURN
1336 002710      101$:    MSG$R      1F      ; ILLEGAL INTERRUPT PRIORITY VALUE
1337
1338
1339      .IF DF      R$MPL      ; [MP01]
1340      C.URM:      MOV      .PNUMB,$URM      ; STORE UNIBUS RUN MASK      ; [MP01]
1341      RETURN      ; [MP01]
1342
1343      C.RCD1:      MOV      .PCHAR,$RCDEV      ; STORE FIRST CHAR OF RECONFIG. DEVICE ; [MP01]
1344      RETURN      ; [MP01]
1345
1346      C.RCD2:      MOV      .PCHAR,$RCDEV+1 ; STORE SECOND CHAR OF RECON. DEVICE ; [MP01]
1347      RETURN      ; [MP01]
1348
1349      .ENDC      ; [MP01]
1350

```

VCFG
SYMBOL
SYMBOL
ADDDF
AD.NUM
BLKSMN
BLKSMX
CERR

CFERR

CFGBF
CFGEXT
CFGFLG
CFGKW
CFGNAM
CFGST
CFGSZ
CFLIN

CHERR
CHKBLK
CHKWND
CHNLMX
CH.SYN
CL.OPN
CNTDF
CTL
C.CSR
C.CTL
C.PRI
C.VECT
C1.BSY
C1.DCP
C1.SDL
C1.X25
DDMDF
DE.OFF
DE.ON
DTEDES
DTEFLG
D.EXT
D.NAME
ERRBUF
ERRMSG
FEADF
FG.X2P
FG.X3P
FL.FDX
FL.HDX
FL.KMX

VCFG CREATED BY MACRO ON 15-JUL-85 AT 19:06 PAGE 1 D 7
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
ADDDF	000342 R	11-523
AD.NUM	002076 R	#25-1186
BLKSMN	= 000040	35-1662
BLKSMX	= 002000	#6-90 35-1664
CERR	001234 RG	7-180 7-181 7-182 7-183 7-184 7-185 7-186 7-187 7-188 7-189 7-190 7-191 7-192 7-196 7-197 7-204 7-205 7-206 7-207 7-208 7-209 7-210 7-211 7-212 7-216 7-217 7-218
CFERR	001244 RG	#12-566 7-193 7-194 7-195 7-198 7-199 7-200 7-201 7-202 7-203 7-213 7-214 7-215 #12-569 16-709
CFGBF	002472 RG	#7-232 10-480 11-537
CFGEXT	000006 R	7-119 9-457
CFGFLG	000136 RG	#7-137 8-365 8-395 8-410 8-420 *11-519
CFGKW	000000 RG	11-535 #17-717
CFGNAM	000000 R	#7-118 9-456
CFGST	000000 RG	#17-717
CFGSZ	002700 RG	#7-233 *10-484 11-536 16-710
CFLIN	001476 RG	7-180 7-181 7-182 7-183 7-184 7-185 7-186 7-187 7-188 7-189 7-190 7-191 7-192 7-195 7-196 7-197 7-204 7-205 7-206 7-207 7-208 7-209 7-210 7-211 7-212 7-216 7-217 7-218
CHERR	003060 R	28-1370 #28-1387 28-1393 28-1409
CHKBLK	004172 RG	#35-1659
CHKWND	004230 RG	#35-1673
CHNLMX	= 007777 G	#6-93
CH.SYN	= 000004	29-1473
CL.OPN	= ***** GX	8-262
CNTDF	000456 R	11-527
CTL	000015 R	#7-125 *8-270 25-1129
C.CSR	002522 R	#27-1297
C.CTL	001706 R	#25-1128
C.PRI	002626 R	#27-1324
C.VECT	002433 R	#27-1280
C1.BSY	= 000002	#6-101
C1.DCP	= 000001	29-1454
C1.SDL	= 000003	#6-100
C1.X25	= 000004	#6-102
DDMDF	000372 R	#6-103
DE.OFF	= 000002 G	11-525
DE.ON	= 000001 G	#7-146
DTEDES	000140 RG	#7-145
DTEFLG	000142 RG	#7-143
D.EXT	001624 R	#7-144
D.NAME	001512 R	#25-1104
ERRBUF	000027 R	#25-1072
ERRMSG	000017 R	#7-133 12-570
FEADF	000662 R	#7-131 12-582
FG.X2P	= 000001 G	11-531
FG.X3P	= 000002 G	#7-138 8-365 11-519
FL.FDX	= ***** GX	#7-139 8-395 8-410 8-420
FL.HDX	= ***** GX	28-1376
FL.KMX	= ***** GX	28-1379 30-1508

```

102          .SBTTL  LOCAL DATA
103          ***
104          : LOCAL DATA
105          ***
106
107 000000          .PSECT  DATA,D
108
109          .NLIST  BEX
110
111
112          :
113          : CONFIG FILE NAME AND EXTENSION
114
115 000000          103      105      124  CFGNAM: .ASCIIZ  "CETAB"
116 000006          115      101      103  CFGEXT: .ASCIIZ  "MAC"
117
118          :
119          : ERROR MESSAGE BUFFER
120
121          :
122 000012          015      116      120  ERRMSG: .BYTE    15
123 000013          126      116      120  :          .ASCII  "VNP -- "
124 000022          :          .BLKB    70.
125          :
126          .EVEN
127
128          :
129          : RECORD BUFFER AND BUFFER LENGTH
130
131 000130          CFGBF: .BLKB    134.
132 000336          CFGSZ: .BLKW    1
133
134          :
135          : SAVED STACK POINTER FOR ERROR EXIT
136
137 000340          SPSAV: .BLKW    1
138
139          :
140          : SYNTAX ERROR FLAG
141          :
142 000342          SYNERR: .BLKW    1
143 000344          MRKFL: .BLKW    1          ; MARKED-FOR-LOAD FLAG
144
145 000346          LINNAM: .BLKW    1          ; DEVICE NAME FROM LINE-ID (RAD50)
146 000350          LINCTL: .BLKB    1          ; CONTROLLER NUMBER
147 000351          LINUNT: .BLKB    1          ; UNIT NUMBER
148 000352          NEXT: .BLKW    1          ; ADDRESS OF NEXT AVAILABLE BYTE
149 000354          PORTNO: .BLKW    1          ; PORT NUMBER FROM PSN$DF MACRO
150 000356          COUNT: .BLKB    1          ; COUNT OF NUMBER OF DIGITS IN DTE ADDRESS
151 000357          CVTBUF: .BLKB    1          ; CONVERSION BUFFER FOR ASCII TO DECIMAL DIGIT
152          .EVEN
153
154          :
155          : DATA FOR X3P$DF MACRO
156 000360          BLKSZD: .BLKW    1          ; DEFAULT BLOCK SIZE
157 000362          BLKSZM: .BLKW    1          ; MAXIMUM BLOCK SIZE
158 000364          WNDSDZ: .BLKW    1          ; DEFAULT WINDOW SIZE

```

577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612

```

.SBTTL PCKBCD - PACK STRING OF DIGITS IN BCD FORMAT
*
PCKBCD - PACK A STRING OF DIGITS IN BCD FORMAT
INPUTS:
R0 - NUMBER OF BYTES IN PACKED ADDRESS
R4 - ADDRESS OF BUFFER CONTAINING ASC: DIGITS TO PACK (0=END OF BUFFER)
R5 - ADDRESS OF BUFFER TO STORE PACKED DATA
COUNT - NUMBER OF DIGITS SPECIFIED
OUTPUTS:
BUFFER SPECIFIED IN R5 CONTAINS DATA IN BCD FORMAT
-
PCKBCD::MOV R5,R1 ; GET ADDRESS OF PACKED DATA
MOV COUNT,R2 ; GET NUMBER OF DIGITS IN STRING
BEQ 40$ ; BR IF NOTHING TO DO
5$: CLRB (R1)+ ; INITIALIZE PACKED ADDRESS
DEC R0 ; MORE TO INITIALIZE?
BGT 5$ ; BR IF YES
CLR R1 ; POSITION INDICATOR (0=HIGH FOUR BITS)
DEC R5 ; UPDATE ADDRESS FOR FIRST TIME
10$: MOVB (R4)+,R0 ; GET DIGIT TO PACK
TST R1 ; STORE IN HIGH FOUR BITS?
BNE 20$ ; BR IF NO
ASL$ 4,R0 ; MOVE TO HIGH FOUR BITS
INC R1 ; INDICATE STORE IN LOW FOUR BITS NEXT
INC R5 ; POINT TO NEXT BYTE
BR 30$ ; BR TO STORE DECIMAL
20$: DEC R1 ; INDICATE STORE IN HIGH FOUR BITS NEXT
30$: BISB R0,(R5) ; PACK THE NUMBER INTO THE BUFFER
DEC R2 ; MORE TO CONVERT?
BGT 10$ ; BR IF YES
40$: RETURN

```

000356'

VCFP CREATED BY MACRO ON 29-JUN-85 AT 02:18 PAGE 4 D 10
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
SF.ACT	= 000200	#5-67
SF.ENA	= 000100	#5-67
SF.LPB	= 000004	#5-67
SF.MFL	= 000040	#5-67
SF.PAC	= 000020	#5-67
SF.REA	= 000010	#5-67
SF.SER	= 000001	#5-67
SF.SVC	= 000002	#5-67
SF.UNL	= 000040	#5-67
SPACE	= 000040	#6-88
SPFILL	001444 RG	19-734
SPSAV	000340 R	#7-137
STORID	001324 RG	#19-729
STRNXT	001254 RG	#18-687
SUBAMN	= 000000 G	#6-93
SUBAMX	= 023417 G	#6-94
SYNERR	= 000342 RG	#7-142
SYSFDB	= ***** GX	*16-628
S\$BAS	= *****	*16-633
		8-259
		8-259
		8-264
		8-264
		8-265
		8-265
		8-266
		8-266
		8-270
		8-270
		8-275
		8-275
		8-262
		8-262
		8-267
		8-267
		8-271
		8-271
		8-272
		8-272
		8-276
		8-276
S.COST	000001	#5-67
S.FLG	000000	#5-67
S.LEN	000004	#5-67
S.NMST	000002	#5-67
S.OWNR	000003	#5-67
WNDSMX	= 000177	#6-95
WNDSZD	000364 RG	#7-158
WNDSZM	000366 RG	#7-159
Z.NAM	= ***** GX	17-661
\$AZFS	= ***** GX	10-422
\$BFR	= ***** GX	16-630
\$CAT5	= ***** GX	14-527
\$CD1B	= ***** GX	18-692
\$CEACC	= ***** GX	16-632
\$CLBUF	= ***** GX	*11-444
\$CLDEQ	= ***** GX	9-396
\$CLFLG	= ***** GX	*9-305
\$CLIOS	= ***** GX	10-424
\$CLQUE	= ***** GX	10-423
\$CLSAV	= ***** GX	9-304
\$CLSBK	= ***** GX	9-393
\$CLSIZ	= ***** GX	*11-445
\$DBG\$	= *****	9-303
\$DFUIC	= ***** GX	10-419
\$ERRMA	00152 RG	#8-271
\$ERRMB	00151 RG	#8-272
\$ERRM4	00141 RG	#8-268
\$ERRM5	00141 RG	#8-269
\$ERRM6	00141 RG	#8-270
\$ERRRO	0011 RG	#8-262

VCIO
 SYMEOL
 SYMBOL
 DSKBUF
 DSEND
 DSHIOR
 DERN
 ERFOR
 ERR10
 ERR12
 ERR14
 ERR16
 ERR18
 ERR2
 ERR20
 ERR22
 ERR24
 ERR26
 ERR28
 ERR30
 ERR32
 ERR34
 ERR36
 ERR4
 ERR6
 ERR8
 F.RSI
 L\$BEX
 L\$BFL
 L\$BLD
 L\$BMX
 PR.LBI
 PR.XFI
 R\$SEI
 SYSFD
 T\$NHI
 \$ALL1
 \$ALL1
 \$BFR
 \$CLDE
 \$CLIO
 \$CLOP
 \$CTBI
 \$DECI
 \$DECP
 \$IOSB
 \$LBN
 \$LLEN
 \$PUTR
 \$RADD
 \$READ
 \$RLBL
 \$SAVV

VCIO CREATED BY MACRO ON 29-JUN-85 AT 02:18 PAGE 1
 SYMEOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
DSKBUF	000000 R	#6-125 8-189
DSEND	= 000104	10-279 10-302
D\$HIOR	000024	*10-297 *10-298 *10-299 *10-300
D\$RNN	000002	10-294 *10-295
ER\$POR	000000 R	#6-118 *7-147 7-157 7-160 *7-166 *7-168 *8-216 *8-218 *8-220
		*8-222 *8-224 *9-261
ERR10	000012	#5-95 8-218
ERR12	000014	#5-94
ERR14	000016	#5-95 8-222
ERR16	000020	#5-96 8-224
ERR18	000022	#5-97
ERR2	000002	#5-89 7-166
ERR20	000024	#5-98
ERR22	000026	#5-99
ERR24	000030	#5-100
ERR26	000032	#5-101
ERR28	000034	#5-102
ERR30	000036	#5-103
ERR32	000040	#5-104
ERR34	000042	#5-105 9-261
ERR36	000044	#5-106 8-220 10-283
ERR4	000004	#5-90
ERR6	000006	#5-91 7-168
ERR8	000010	#5-92 8-216
F\$RSIZ	= ***** GX	*9-251 *10-302
L\$BEXT	000352	8-199
L\$BFLG	000030	8-192
L\$BLDZ	000016	8-194
L\$BMXZ	000020	8-195
PR.LBL	000076 R	7-153 #8-189
PR.XFR	000272 R	7-155 #9-239
R\$EIS	= *****	8-203
SYSFDB	= ***** GX	*9-251 *10-302
TS\$NHD	= 040000	8-192
\$ALL15	= ***** GX	10-280
\$ALL16	= ***** GX	8-207
\$BFR	= ***** GX	9-243 10-287 10-291
\$CLDEQ	= ***** GX	7-156
\$CLOS	= ***** GX	7-149 8-215 9-260
\$CLOPE	= ***** GX	7-148
\$CTBIO	000030 RG	#7-146
\$DECHM	000416 RG	#10-279
\$DECP	= ***** GX	*10-286 10-292 10-302
\$IOSB	= ***** GX	8-215 9-260
\$LBN	= ***** GX	*9-249 *9-250
\$LEN	= ***** GX	*8-204 9-241 *9-242 *9-246 9-251 9-252
\$PUTRC	= ***** GX	9-251 10-302
\$RADDR	= ***** GX	*9-251 *10-302
\$READ	= ***** GX	9-247
\$RLBL	= ***** GX	8-190
\$SAVVR	= ***** GX	9-240

VCLOQR
 SYMBO
 SYMBO
 LO.SE
 LO.SK
 LO.SK
 LO.SN
 LO.TP
 LO.TR
 LO.TR
 LO.UN
 LO.VC
 LO.VE
 LR.AC
 LR.AC
 LR.AD
 LR.AL
 LR.BL
 LR.CT
 LR.DE
 LR.HL
 LR.LI
 LR.NA
 LR.PA
 LR.PR
 LR.ST
 LR.TR
 LR.TY
 LR.UC
 LR.UJ
 LR.UN
 LS.AL
 LS.CE
 LS.CI
 LS.CX
 LS.CX
 LS.EC
 LS.FD
 LS.HD
 LS.HL
 LS.LI
 LS.LM
 LS.LC
 LS.MC
 LS.NC
 LS.NI
 LS.OE
 LS.OF
 LS.PF
 LS.TC
 LS.TS
 LS.UN
 LS.XS
 LX.AL
 LX.CE

VCLQRM CREATED BY MACRO ON 29-JUN-85 AT 02:18 PAGE 2 D 12
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
LO.SEG	000054	#5-59
LO.SKA	000014	#5-59
LO.SKN	000014	#5-59
LO.SNM	000035	#5-59
LO.TPA	000020	#5-59
LO.TRB	000034	#5-59
LO.TRI	000030	#5-59
LO.UNT	000033	#5-59
LO.VCT	000014	#5-59
LO.VER	000042	#5-59
LR.ACC	000020	#5-59
LR.ACO	000052	#5-59
LR.ADD	000012	#5-59
LR.ALI	000002	#5-59
LR.BLK	000010	#5-59
LR.CTL	000004	#5-59
LR.DES	000010	#5-59
LR.HLP	000002	#5-59
LR.LIN	000002	#5-59
LR.NAM	000012	#5-59
LR.PAS	000041	#5-59
LR.PRO	000002	#5-59
LR.STS	000000	#5-59
LR.TRI	000006	#5-59
LR.TYP	000002	#5-59
LR.UCB	000016	#5-59
LR.UIC	000020	#5-59
LR.UNT	000005	#5-59
LS.ALI	= 000016	#5-59
LS.CEX	= 000001	#5-59
LS.CIR	= 000033	#5-59
LS.CXO	= 010000	#5-59
LS.CXS	= 000400	#5-59
LS.ECH	= 001000	#5-59
LS.FDX	= 004000	#5-59
LS.HDX	= 010000	#5-59
LS.HLP	= 000012	#5-59
LS.LIN	= 000003	#5-59
LS.LMC	= 000007	#5-59
LS.LOG	= 000020	#5-59
LS.MOD	= 000036	#5-59
LS.NOD	= 000014	#5-59
LS.NTI	= 000013	#5-59
LS.OBJ	= 000022	#5-59
LS.OPT	= 000400	#5-59
LS.PRO	= 000002	#5-59
LS.TCP	= 002000	#5-59
LS.TS	= 000010	#5-59
LS.UNF	= 020000	#5-59
LS.XIT	= 000011	#5-59
LX.ALI	= 000017	#5-59
LX.CEX	= 000004	#5-59

5-59 5-59 5-59 5-59 5-59 5-59

7-106

VDEV
FIND


```

264 .SBTTL FIND - FIND DATA STRUCTURES
265
266 ;+ FIND - DETERMINE IF DATA STRUCTURES ALREADY EXIST
267
268 ; THIS ROUTINE IS CALLED TO LOOK FOR AN EXISTING DEVICE WHOSE NAME MATCHES
269 ; $$DEV. IF SUCH A DEVICE IS LOCATED, ITS DCB ADDRESS IS STORED IN FOUND.
270
271 ; INPUTS:
272 ; $$DEV=2 CHARACTER DEVICE NAME
273
274 ; OUTPUTS:
275 ; FOUND=DCB ADDRESS (IF SUCCESSFUL)
276
277 000242 005000 FIND: CLR R0 ; INDICATE NO DCB (YET)
278 000244 016767 000000G 000204' 10$: MOV $DEVHD,DEVBUF ; GET DEVICE POINTER
279 000252 016767 000204' 000000G 10$: MOV DEVBUF,$RADDR ; GET NEXT DEVICE IN LIST
280 000260 001413 BEQ 30$ ; IF EQ, END OF LIST
281 000262 GETRV #D.PCB+2 ; READ IN DEVICE CONTROL BLOCK
282 000274 026767 000000G 000000C CMP $$DEV,D.NAM+DEVBUF ; DOES DEVICE NAME MATCH?
283 000302 001363 BNE 10$ ; IF NE, NO
284 000304 016700 000000G MOV $RADDR,R0 ; LOAD DEVICE POINTER INTO R0
285 000310 000241 30$: CLC ; NO ERRORS YET
286 000312 010067 000166' MOV R0,FOUND ; IS THERE ALREADY SUCH A DEVICE?
287 000316 001416 BEQ 40$ ; IF EQ, NO
288 000320 005767 000000C TST D.DSP+DEVBUF ; IS THE DRIVER RESIDENT?
289 000324 001014 BNE 101$ ; IF NE, YES - VERY BAD
290 000326 005001 CLR R1 ; GET FIRST UNIT #
291 000330 156701 000000C BISB D.UNIT+DEVBUF,R1 ; ...
292 000334 005002 CLR R2 ; AND LAST UNIT #
293 000336 156702 000000C BISB D.UNIT+DEVBUF,R2 ; ...
294 000342 160102 SUB R1,R2 ; CALCULATE UCB COUNT
295 000344 005202 INC R2 ; ...
296 000346 003406 BLE 111$ ; IF LE, D.UNIT IS SCREWED
297 000350 010267 000172' 40$: MOV R2,UNITS ; SAVE UNIT COUNT
298 000354 RETURN
299
300 ; ERROR CONDITION
301
302 101$: MSG$R 2X ; DEVICE DRIVER ALREADY RESIDENT!
303 000356 111$: MSG$R 2Q ; ILLEGAL UNIT (UCB) COUNT
304 000364

```

ASSCHK= 000000	FMT9 003532R	002 R\$S11M= 000000	S4.DEC= 004000	US.MDM= 000020
ASSCPS= 000000	FM.4 = 000000	STBEXT 000000R	S4.DLO= 000100	US.MNT= 000100
ASSPRI= 000000	FM.4A = 000000	SYSFDB= ***** GX	S4.EDT= 010000	US.OFL= 000001
ASSTRP= 000000	FM.6 = 000000	S\$SWRG= 000000	S4.HFF= 000020	US.PUB= 000004
BASE 000170R	FM.9 = 000000	S\$SYZ= 007600	S4.HFL= 000007	US.PWF= 000010
C\$SCKP= 000000	002 FOUND 000166R	002 S.LHD= ***** GX	S4.HHT= 000040	US.RED= 000002
C\$SORE= 000400	F\$SLVL= 000001	S1.DEC= 001000	S4.HSY= 000200	US.SHR= 000001
C\$SRSH= 177564	F.NRBD= ***** GX	S1.DPR= 000400	S4.RGS= 020000	US.SPU= 000002
DBLK1 000126R	002 F.RSIZ= ***** GX	S1.DSI= 004000	S4.SFC= 040000	US.VV= 000001
DBLK2 000140R	002 F.URBD= ***** GX	S1.ESC= 000002	S4.VFL= 000010	US.WCK= 000010
DCB 000150R	002 G\$STPP= 000000	S1.IBF= 002000	S5.ABP= 100000	UU.ABD= 000400
DEVBUF 000204R	002 G\$STSS= 000000	S1.IBY= 000200	S5.BCC= 020000	UU.ATN= 000100
DNAME 000004R	002 G\$STTK= 000000	S1.OBY= 000100	S5.DAO= 040000	UU.AVN= 000004
DSKBUF 000000R	003 G\$SWRD= 000000	S1.PTH= 000010	S5.HPC= 000014	UU.GUS= 000010
DUMMY 000152R	002 I\$SRAR= 000000	S1.RES= 010000	S5.HPD= 000020	UU.IOS= 002000
DVALL 000772R	I\$SRDN= 000000	S1.RNE= 000020	S5.ITI= 000100	UU.ONL= 000020
DVDEA 002142R	K\$CNT= 177546	S1.RNF= 020000	S5.OXF= 000040	UU.RCT= 000002
DVERR 002054R	K\$CSR= 177546	S1.RSP= 000004	S5.RPO= 002000	UU.RDY= 000200
DVLBL 000632R	K\$SLDC= 000000	S1.RST= 000001	S5.SW1= 000001	UU.SER= 000001
DVSYM 000372R	K\$STPS= 000074	S1.TNE= 040000	S5.TMM= 000002	UU.SID= 001000
DVXFR 001116R	LD\$LP= 000000	S1.TSY= 000040	S5.VER= 010000	UU.SPC= 000040
DV.CCL= 000002	LENGTH 000136R	002 S1.USI= 100000	S5.XDF= 000004	U.ACAB 000062
DV.COM= 000000	L\$BFLG= ***** GX	S2.BEL= 020000	S5.XON= 000010	U.ACP= 000032
DV.DIR= 000010	L\$BLDZ= ***** GX	S2.BRO= 000020	S6.EIO= 000400	U.ADMA 000066
DV.EXT= 000400	L\$BMXZ= ***** GX	S2.CR= 002000	S6.RDI= 100000	U.AFLG 000064
DV.F11= 040000	L\$BSA= ***** GX	S2.CTD= 040000	S6.RLU= 001000	002 U.ATT 000022
DV.ISP= 002000	L\$ASG= 000000	S2.CTS= 100000	TABEND 000202R	U.BPKT= 000044
DV.MBC= 000400	L\$DRV= 000000	S2.ELF= 001000	TS\$NHD= ***** GX	U.BUF= 000024
DV.MNT= 100000	L\$SP11= 000001	S2.FLF= 000400	TS\$KMG= 000000	U.CBF= 000032
DV.MSD= 000100	L\$11R= 000000	S2.IRQ= 000200	TS\$MIN= 000000	U.CLI 177772
DV.OSP= 004000	M\$SCRB= 000124	S2.DBF= 004000	UC.ALG= 000200	U.CNT 000030
DV.PSE= 010000	M\$SCRX= 000000	S2.DRQ= 002100	UC.ATT= 000010	U.COTQ 000030
DV.REC= 000001	M\$SFC= 000000	S2.PCU= 010000	UC.KIL= 000004	U.CTCB 000026
DV.SDI= 000020	M\$MGE= 000000	S2.RCU= 000001	UC.LGH= 000003	U.CTL 000004
DV.SOD= 000040	M\$NET= 000000	S2.SRQ= 000040	UC.NPR= 000100	U.CTYP 000052
DV.SWL= 001000	M\$OVR= 000000	S2.WAL= 000010	UC.PWF= 000020	U.CW1 000010
DV.TTY= 000004	N\$ACC= 000001	S2.WRA= 000006	UC.QUE= 000040	U.CW2 000012
DV.UMD= 000200	N\$SBUF= 000001	S2.WRB= 000002	UD.UNS= 000000	U.CW3 000014
D\$BUG= 177514	N\$LDV= 000001	S3.ACR= 000001	UD.160= 000004	U.CW4 000016
D\$ISK= 000000	N\$MCP= 000001	S3.CTC= 000004	UD.200= 000001	U.CYL= 000104
D\$SL11= 000001	N\$MLL= 000001	S3.FDX= 000200	UD.556= 000002	U.DCB 000000
D\$SYNC= 000000	N\$MOV= 000010	S3.ICE= 001000	UD.625= 000005	U.FCDE= 000042
D\$SYNM= 000000	N\$NCT= 000001	S3.MHE= 000400	UD.8K= 000006	U.FNUM= 000040
D.DSP= ***** GX	N\$PEM= 000001	S3.NEC= 000020	UD.800= 000003	U.GRP= 000102
D.NAM= ***** GX	PBUF 000174R	002 S3.PMT= 020000	UM.CLI= 000036	U.KCSR= 000032
D.PCB= ***** GX	PLEN 000176R	002 S3.PTH= 004000	UM.DSB= 000200	U.KCS6= 000034
D.UCB= ***** GX	P\$SP45= 000000	S3.RAL= 000010	UM.NBR= 000400	U.LUIC 177774
D.UCBL= ***** GX	P\$SWRD= 000000	S3.RES= 010000	UM.OVR= 000001	002 U.MEDI= 000040
D.UNIT= ***** GX	Q\$GPT= 000010	S3.RUB= 040000	UNITS 000172R	U.MLUN= 000050
ERRBUF 000017R	002 REP4A 002214R	S3.TAB= 000002	US.ABO= 000001	U.MUP 177772
ERRMSG 000007R	002 REP4A1 002224R	S3.TME= 002000	US.BSP= 000002	U.OWN 177776
ETAB 0000156R	002 REP4A2 002264R	S3.TSY= 000040	US.BSY= 000200	U.RBNS= 000112
E\$XPR= 000000	REP9 002346R	S3.8BC= 000100	US.FDR= 000040	U.RCTC= 000113
FIND 000242R	RVAD 002004R	S4.ABD= 100000	US.FRK= 000002	U.RCTS= 000110
FMT4 003461R	002 R\$SDEF= 000000	S4.ANI= 000400	US.KPF= 000001	U.RED 000002
FMT4A 003420R	002 R\$K11= 000001	S4.AV0= 001000	US.LAB= 000004	U.RED2 000034
FMT6 003511R	002 R\$SND= 000000	S4.BLK= 002000	US.MDE= 000002	

```
219 000652 105 124 110 BROAD: .ASCIZ /ETHERNET/
220 ;
221 ; ASCIZ STRING FOR MODULE HEADER
222 ;
223 000663 115 157 144 MSPEC: .ASCIZ /Module/
224 ;
225 ; ASCIZ STRINGS FOR MODULE ID
226 ;
227 000672 130 062 065 MX25PR: .ASCIZ /X25-PROTOCOL/
228 000707 130 062 065 MX25AC: .ASCIZ /X25-ACCESS/
229 000722 130 062 065 MX25SV: .ASCIZ /X25-SERVER/
230 000735 130 062 071 MX29SV: .ASCIZ /X29-SERVER/
231 ;
232 ;
233 000750 054 040 124 GRPBIL: .ASCIZ /, Type = BILATERAL/
234 ; EVEN
235 ;
236 000774 ; ERMSG$ < No information%N> ; NO INFORMATION ERROR MESSAGE
237 001020 ; ERBLK$ NOINFO
238 ;
239 ; NTINIT TASK NAME IN RAD50
240 ;
241 001024 055251 054374 ; NTINAM: .RAD50 /NTINIT/
242 ;
243 001030 ; PLIPDV: .BLKW 1 ; ADDRESS OF PLI'S PDV INDEX
244 001032 ; NODSAV: .BLKW 1 ; REMOTE NODE UNMAPPED ADDRESS
245 ;
246 ;
247 ; Temporary storage - each field MUST begin on a word boundary ; [TD001]
248 ; ; **=1
249 001034 ; TEMP1: .BLKB 22. ; TEMPORARY STORAGE FOR OUTPUT STRINGS
250 001062 ; TEMP2: .BLKB 80. ;
251 001202 ; TEMP3: .BLKB 18. ; [ room for net. mgmt. 'id-string' ] ; [TD001]
252 001224 ; TEMP4: .BLKB 16. ; **=1
253 001244 ; TEMP5: .BLKB 200. ;
254 001554 ; TEMP6: .BLKW 1. ;
255 001556 ; TEMP7: .BLKW 1. ;
256 001560 ; TEMP8: .BLKB 12. ;
257 ;
258 001574 ; FLAGS: .BLKB 1 ; FLAGS BYTE
259 001575 060 061 062 CVTHEX: .ASCIZ /0123456789ABCDEF/ ; TABLE FOR HEX CONVERSIONS
260 000020 ; MXHEX = -.CVTHEX
261 ; EVEN
262 ; DSABL LC
263 ;
264 ;
265 ; ATTACH AND DETACH TERMINAL DPB'S
266 ;
267 001616 ; TIATT: QIOW$ IO.ATT,2,2 ; ATTACH TERMINAL
268 001646 ; TIDET: QIOW$ IO.DET,2,2 ; DETACH TERMINAL
269 ;
270 ;
271 ; LOGGING SINK TYPE TABLE
272 ;
273 001676 000400 ; LOGTYP: .WORD OP$CON ; LOGGING MONITOR
274 001700 000200 ; .WORD OP$FIL ; LOGGING FILE
275 001702 000100 ; .WORD OP$MON ; LOGGING CONSOLE
```

VDIS - VNP SHOW COMMANDS
\$DLOG - SHOW LOGGING

MACRO V05.03b Monday 15-Jul-85 12:14 Page 12-2

D 16

```
759 ;
760 004056 ERMSG$ <%N%A %A as of %Y%N>
761 004100 ERBLK$ LOMSG1,<TEMP2,TEMP1>
762
763 004110 ERMSG$ < State = %A>
764 004125 ERBLK$ LOMSG2,<TEMP1>
765
766 004132 ERMSG$ < Sink node = %D (%A), events =>
767 004172 ERBLK$ LOMSG3,<TEMP1,TEMP2>
768
769 004202 ERMSG$ < Sink node = host, %D.%D (%A), events =>
770 004254 ERBLK$ LOMSG3X,<TEMP7,TEMP1,TEMP2>
771
772 004266 ERMSG$ < Sink node = %D.%D (%A), events =>
773 004331 ERBLK$ LOMSG3Y,<TEMP7,TEMP1,TEMP2>
774
775
776 004342 ERMSG$ < Sink node = host, %D (%A), events=>
777 004410 ERBLK$ LOMSG3A,<TEMP1,TEMP2>
778
779 004420 ERMSG$ < %A>
780 004430 ERBLK$ LOMSG4,<TEMP2>
781
782 004436 ERMSG$ < >
783 004437 ERBLK$ LOMSG5
784
785 004442 ERMSG$ <%NLogging = %A%N>
786 004462 ERBLK$ LOMSG6,<TEMP1>
787
788 004470 ERMSG$ < Sink name = %A>
789 004511 ERBLK$ LOMSG7,<TEMP2>
790
791 004516 ERMSG$ <Event logging not supported>
792 004551 ERBLK$ LOERR1
793
794 004554 ERMSG$ <Invalid sink node>
795 004575 ERBLK$ LOERR2
796
797 .DSABL LC
```

E 16

Page 13

```
74          .SBTTL  LOCAL DATA
75          ;***
76          ; LOCAL DATA
77          ;***
78
79 000000          .PSECT  DATA,D
80
81          .NLIST  BEX
82
83
84          ;
85          ; LOCAL EQUIV'S
86          Ticks  = 1
87
88
89          ;
90          ; ERROR MESSAGES
91          ;
92
93 000000          ERMSG$ <VNP - NTINIT not installed>
94 000032          ERBLK$ ERR1
95 000036          ERMSG$ <VNP - NTINIT not fixed>
96 000064          ERBLK$ ERR2
97 000070          ERMSG$ <VNP - NTINIT send data packet allocation failure>
98 000150          ERBLK$ ERR3
99 000154          ERMSG$ <VNP - NTINIT clock queue entry failure>
100 000222          ERBLK$ ERR4
101 000226          ERMSG$ <VNP - NTINIT active>
102 000251          ERBLK$ ERR5
103
104 000254          MSGB:  .BLKB  80.          ; MESSAGE BUFFER
105
106          ;
107          ; COMPATIBLE SYMBOLS
108          ;
109 000374          $PRCLN::
110 000374          $PREXT::
111 000374          $PRIN::
112 000374          $PRLEN::
113 000374          $PRLN::
114 000374          .BLKW  1          ; PROCESS INCREMENT
115
116          ;
117          ; COMM EXEC FILE NAME
118 000376          103      105      130  CEXNAM: .ASCIZ  'CEX'
119
120          .IF  DF  R$$MPL
121
122          RSXNAM: .ASCIZ  'RSX11M'
123          .EVEN
124
125          VEXNAM: .ASCIZ  'RSXVEC'
126          .EVEN
127
128          .ENDC
129
130          ;
```

```
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669 001206 01270' 000020
670 001212
671 001216 103516
672
673 001220 010046
674 001222 010546
675 001224 010446
676
677 001226 012704 000000G
678 001232 012705 000410'
679
680 001236 012665 000006
681 001242 012765 000004 000002
682 001250 005065 000012
683 001254 005065 000014
684 001260 012665 000004
685 001264
686 001304 061465 000006
687 001310 005565 000010
688
689
690
691 001314 016702 000000G
692 001320
693
694 001336 010203
695 001340 011402
696 001342 001421
697 001344
698 001354 026564 000004 000004
699 001362 001435
700 001364 026564 000010 000010
701 001372 101361
702 001374 103404
703 001376 026564 000006 000006
704 001404 101354
705
706 001406 010215

**--RUN-RUN TASK

INPUTS:
R4 - NUMBER OF TICKS TO DELAY FROM SYSTEM BOOT
R5 - TCB ADDRESS ( IN SYSTEM IMAGE )

OUTPUTS:
C-BIT SET IS NO NODE AVAILABLE
CLOCK QUEUE ENTRY GENERATED

NOTE: THIS ALGORITHM WILL RUN THE TASK BEFORE ANY OTHER TASK
      SCHEDULED TO RUN WITH THE SAME DELAY FACTOR,
      ( UNLIKE VMR RUN COMMAND ).

RUN:  MOV    #C.LGTH,R1      ; SIZE OF CLOCK QUEUE ENTRY
      CALL   $ALL15         ; ALLOCATE SPACE FROM RSX11M POOL
      BCS    40$            ; NO SPACE AVAILABLE

      MOV    R0,-(SP)        ; SAVE NODE ADDRESS
      MOV    R5,-(SP)        ; AND TCB ADDRESS
      MOV    R4,-(SP)        ; AND # OF TICKS

      MOV    #SBFR,R4        ; R4 POINTS AT VIRTUAL I/O BUFFER
      MOV    #CLKQB,R5       ; R5 AT CLOCK QUEUE ENTRY

      MOV    (SP)+,C.TIM(R5) ; SET NUMBER OF TICKS
      MOV    #C.SSHT,C.RQT(R5) ; SINGLE SHOT
      CLR    C.RSI(R5)       ; NO RESCHEDULE INT. COUNT
      CLR    C.RSI+2(R5)     ; NO RSI UNITS
      MOV    (SP)+,C.TCB(R5) ; SET TCB ADDRESS
      GETRV   ,ABTIM,#2      ; GET ABSOLUTE TIME COUNTER
      ADD    (R4),C.TIM(R5)  ; FORM ABSOLUTE TIME OF REQUEST
      ADC    C.TIM+2(R5)     ; ...

      ; INSERT NEWLY FORMED ENTRY INTO CLOCK QUEUE IN TEMPORAL SEQUENCE

      MOV    ,CLKHD,R2       ; PREVIOUS ENTRY
      GETRV   R2,#C.LGTH     ; READ IN LISTHEAD

10$:  MOV    R2,R3            ; SAVE PREVIOUS ENTRY ADDRESS
      MOV    (R4),R2         ; GET ADDRESS OF NEXT ENTRY
      BEQ    30$            ; END OF LIST
      GETRV   R2             ; READ IN NEXT CLOCK QUEUE ENTRY
      CMP    C.TCB(R5),C.TCB(R4) ; TASK ALREADY ACTIVE ?
      BEQ    101$           ; YES
      CMP    C.TIM+2(R5),C.TIM+2(R4) ; NEW ENTRY AT LATER TIME ?
      BHI    10$            ; NO
      BLO    30$            ; YES
      CMP    C.TIM(R5),C.TIM(R4) ; MAYBE
      BHI    10$            ; PREPT IF TIME IS SAME

30$:  MOV    R2,(R5)         ; POINT OUR NEW ENTRY AT NEXT
```

VCEX
SYMBOL
SYMBOL
HEADR
LLCTB
NETPF
NTGCD
NTPCB
NTUMR
PDVTA
PDVTB
PLGTH
SBPCB
SLTMB
STKDP
TLGTH
TSKHD
..COO

VCEX CREATED BY MACRO ON 29-JUN-85 AT 02:14 PAGE 5 E 3

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
.HEADR	= ***** GX	6-170
.LLCTB	= 000660 R	6-219 #6-222 11-567
.NETPF	= ***** GX	8-350
.NTGCD	= 003000	#5-65
.NTPCB	= 000000	#5-65
.NTUMR	= 000002	#5-65
.PDVTA	= ***** GX	11-581
.PDVTB	= 000634 R	#6-216 9-454 11-565
.PLGTH	= ***** GX	7-318
.SBPCB	= 000066	#5-65
.SLTMB	= 000646 R	6-216 #6-219 11-566
.STKDP	= ***** GX	6-173
.TLGTH	= ***** GX	12-636 14-786
.TSKHD	= ***** GX	12-636
..COO	= ***** GX	14-815

VCFG -
\$NLCFG

418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450

```

418 ; END OF FILE FOUND (NO X3P$DF OR FEA$DF)
419
420 000612 032767 000002 000136' 188$: BIT #FG,X3P,CFGFLG ; IS X3P$DF NEEDED?
421 000620 001032 BNE 161$ ; BR IF YES - ERROR
422
423 ; ALL NEEDED PARAMETERS SEEN
424
425 000622 126727 000000G 000000G CMPB $CLSBK,#IE.EOF ; EOF ?
426 000630 001402 BEQ 200$ ; IF YES - BRANCH
427
428 000632 190$: CALL $CLDEQ ; CLOSE CONFIG FILE
429 000636 000241 200$: CLC ; INDICATE SUCCESS
430 000640 RETURN
431
432 ;
433 ; ERROR CONDITIONS
434
435 000642 101$: MSG$R 1S ; UNT$DF MISSING
436 000650 111$: MSG$R 1U ; SLT$DF MISSING (EOF)
437 000656 121$: MSG$R 1V ; CNT$DF MISSING (EOF)
438 000664 131$: MSG$R 1W ; UNT$DF MISSING (EOF)
439 000672 141$: MSG$R 1X ; STA$DF MISSING (EOF)
440 000700 151$: MSG$R 1Z ; LLC$DF MISSING (EOF)
441 000706 161$: MSG$R 3A ; X3P$DF MISSING (EOF)
442 000714 171$: MSG$R 3B ; X2P$DF MISSING (EOF)
443 000722 181$: MSG$R 3C ; SVC$DF MISSING (EOF)

```

835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851

[illegible]

E 6

```

1352
1353
1354
1355
1356 002716 005767 000000G U.UNIT: TST .PNUMH ; MAKE SURE IT'S A VALID UNIT NUMBER
1357 002722 001011 BNE CHERR ;
1358 002724 026727 000000G 000060 CMP .PNUMB,#48. ; MAX OF 48. ALLOWED
1359 002732 103005 BHIS 101$ ;
1360 002734 026767 000000G 000152' CMP .PNUMB,UCNT ; IS THIS THE NEXT ASCENDING UNIT NUMBER?
1361 002742 001004 BNE 111$ ; IF NE, NO
1362 002744 RETURN ;
1363 002746 101$: MSG$R 1G ; ILLEGAL UNIT NUMBER
1364 002754 111$: MSG$R 1H ; UN$DF OUT OF ORDER
1365
1366
1367
1368
1369 002762 005767 000000G U.CHAO: TST .PNUMH ; MUST BE SINGLE PRECISION
1370 002766 001034 BNE CHERR ;
1371 002770 016700 000152' MOV UCNT,R0 ; GET UNIT COUNT
1372 002774 016701 000000G MOV $SLTA,R1 ; SLT ADDRESS
1373 003000 016702 000000G MOV .PNUMB,R2 ; AND CHARACTERISTICS VALUE
1374 003004 126100 000000G CMPB L,UNT(R1),R0 ; IS THIS THE KEY UNIT?
1375 003010 001016 BNE 10$ ; IF NE, NO
1376 003012 032767 000000G 000000G BIT #FL.FDX,$FLAGS ; FULL-DUPLEX SPECIFIED ?
1377 003020 001402 BEQ 2$ ; NO
1378 003022 042702 000001 BIC #1,R2 ; YES.. CLEAR BIT 0
1379 003026 032767 000000G 000000G 2$: BIT #FL.HDX,$FLAGS ; HALF-DUPLEX SPECIFIED ?
1380 003034 001402 BEQ 5$ ; NO
1381 003036 052702 000001 BIS #1,R2 ; YES.. SET BIT 0
1382 003042 010267 000000G 5$: MOV R2,$$DCHA ; SAVE DLC CHARACTERISTICS
1383 003046 006300 10$: ASL R0 ; CONVERT TO A DOUBLEWORD INDEX
1384 003050 006300 ASL R0 ;
1385 003052 010260 000004G MOV R2,$$DCHA+4(R0) ; STORE PARAMETER VALUE
1386 003056 RETURN ;
1387 003060 CHERR: MSG$R 1K ; ILLEGAL PARAMETER VALUE
1388
1389
1390
1391
1392 003066 005767 000000G U.CHAT: TST .PNUMH ; MUST BE SINGLE PRECISION
1393 003072 001372 BNE CHERR ;
1394 003074 016700 000152' MOV UCNT,R0 ; GET UNIT COUNT
1395 003100 003100 CALL SETTMO ; SET UP TIMEOUT VALUE
1396 003104 016701 000000G MOV $SLTA,R1 ; AND SLT ADDRESS
1397 003110 126100 000000G CMPB L,UNT(R1),R0 ; IS THIS THE KEY UNIT?
1398 003114 001003 BNE 10$ ; IF NE, NO
1399 003116 016767 000000G 000002G MOV .PNUMB,$$DCHA+2 ; SAVE DLC CHARACTERISTICS
1400 003124 006300 10$: ASL R0 ; CONVERT TO A DOUBLEWORD INDEX
1401 003126 006300 ASL R0 ;
1402 003130 016760 000000G 000006G MOV .PNUMB,$$DCHA+6(R0) ; STORE PARAMETER VALUE
1403 003136 RETURN ;
1404
1405
1406
1407
1408 003140 005767 000000G U.PECH: TST .PNUMH ; MUST BE SINGLE PRECISION

```

VCFG
SYMBOL
SYMBOL
FL.LMC
FMT8
FMT8B
FM.8
FM.8B
FNDP
F.CH1
F.CH2
F.FEA
F.MOD
F.RSIZ
F.SET
GETCSR
HSHADD
HSHSZ
IE.EOF
IE.RBG
I\$AS
LLCDF
LNKEND
L.CH1
L.CH2
L.COST
L.CTL
L.DDM
L.DLC
L.MPF
L.NAME
L.NSTA
L.UNIT
MAXCST
MAXID
MAXNDC
MAXWND
MOVE
MS.CIR
MS.CSR
MS.EXT
MS.INC
MS.LLC
MS.MTP
MS.MUX
MS.PRI
MS.SEC

F 6

VCFG CREATED BY MACRO ON 15-JUL-85 AT 19:06 PAGE 2 E 7

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
FL.LMC	= ***** GX	28-1422 30-1506
FMT8	002422 RG	7-181 7-187 7-191 7-192 7-193 7-194 7-195 7-196 7-197
		7-198 7-199 7-200 7-201 7-202 7-203 7-205 7-206 7-213
FMT8B	002442 RG	7-214 7-215 #7-224 7-224 7-180 7-182 7-183 7-184 7-185 7-186 7-188 7-189 7-190
		7-204 7-207 7-208 7-209 7-210 7-211 7-212 7-216 7-217
FM.8	= 000000	7-218 #7-225 7-181 7-187 #7-191 #7-192 #7-193 #7-194 #7-195 #7-196 #7-197
		7-198 7-199 #7-200 #7-201 #7-202 #7-203 #7-205 #7-206 #7-213
FM.8B	= 000000	7-214 7-215 #7-181 7-182 #7-183 #7-184 #7-185 #7-186 #7-188 #7-189 #7-190
		7-204 7-207 #7-208 #7-209 #7-210 #7-211 #7-212 #7-216 #7-217
FNDPDV	004120 RG	#7-218 #34-1638
F.CH1	003734 R	#32-1571
F.CH2	003756 R	#32-1578
F.FEA	004000 R	#32-1585
F.MOD	004910 R	#32-1590
F.RSIZ	= ***** GX	*33-1612
F.SET	004020 R	#32-1595
GETCSR	002562 R	27-1299 #27-1309 28-1437
HSHADD	000146 RG	#7-148
HSHSZ	000144 RG	#7-147
IE.EOF	= ***** GX	8-425
IE.RBG	= ***** GX	10-495
ISSAS	= *****	27-1332
LLCDF	000724 R	11-533
LNKEND	004030 RG	#33-1612
L.CH1	003714 R	#31-1558
L.CH2	003724 R	#31-1564
L.COST	= ***** GX	*30-1522
L.CTL	= ***** GX	8-270
L.DDM	= ***** GX	29-1470
L.DLC	= ***** GX	29-1470
L.MPF	= ***** GX	25-1242
L.NAME	001512 R	#25-1071
L.NSTA	= ***** GX	25-1235
L.UNT	= ***** GX	8-271
MAXCST	= 000031	28-1374 28-1397 28-1412 29-1456
MAXID	= 000020	#6-87 25-1232 30-1519
MAXNDC	= 000200	#6-84 13-628 13-640
MAXWND	= 000007	#6-88 #6-89
MOVE	001436 RG	13-648 #14-669
MS.CIR	= ***** GX	8-274
MS.CSR	= ***** GX	27-1297 27-1302
MS.EXT	= ***** GX	8-313 25-1107
MS.INC	= ***** GX	8-313 25-1096
MS.LLC	= ***** GX	8-308
MS.MTP	= ***** GX	8-357
MS.MUX	= ***** GX	8-340
MS.PRI	= ***** GX	27-1324 27-1334
MS.SEC	= ***** GX	28-1435

```

159 000366      WNDZM::BLKW 1          ; MAXIMUM WINDOW SIZE
160      ;
161      ; DATA FOR DTE$DF MACRO
162
163 000370      DTEDES::BLKW 1          ; DTE ADDRESS
164 000372      DTEFLG::BLKW 1          ; FLAGS WORD
165      DE.ON == 1          ; SET DTE STATE TO 'ON'
166      DE.OFF == 2         ; SET DTE STATE TO 'OFF'
167      .EVEN
168 000374      HSHSZ::BLKW 1          ; HASH TABLE SIZE
169 000376      HSHADD::BLKW 1          ; ADDRESS OF HASH TABLE
170 000400      CTIM::BLKW 1           ; COUNTER TIMER VALUE
171      ;
172      ; DATA FOR PVC$DF MACRO
173
174 000402      PVSTA::BLKW 1          ; PVC STATE
175      PV.OFF == 1         ; STATE = OFF
176      PV.ON == 2          ; STATE = ON
177      ;
178      ; DATA FOR DST$DF MACRO
179
180 000404      DSFLG::BLKW 1          ; LOCAL FLAG WORD
181      DS.X25 == 1         ; X25 DESTINATION TYPE
182      DS.X29 == 2         ; X29 DESTINATION TYPE
183 000406      DSTTKL::BLKB 1          ; Length of object task name      ;[TD001]
184 000407      DSTTSK::BLKB MAXID      ; Object task name (must follow DSTTSL) ;[TD001]
185 000427      DSTNML::BLKB 1          ; Length of destination name      ;[TD001]
186 000430      DSTNAM::BLKB MAXID      ; Destination name (must follow DSTNML) ;[TD001]
187 000450      DSTPRI::BLKB 1          ; PRIORITY                        ;[TD001]
188 000451      DSTOBJ::BLKB 1          ; OBJECT NUMBER                    ;[TD001]
189      .EVEN
190      ;
191      ; These tables are used to fill in the variable-length fields of
192      ; the destination block. Each entry is the address of a
193      ; buffer containing a count (one byte) followed by the data
194      ; to be stored. The order of the table is the order in which
195      ; fields appear in the destination block. The first table
196      ; (DSTVR0) is used for the new (gateway) fields; the remainder
197      ; describe the old (common) fields.
198      ;
199      ; *** ORDER IS IMPORTANT (SEE DSTDF$ MACRO) ***
200      ;
201 000452      DSTVR0::                ;[TD001]
202 000452 000427' .WORD DSTNML          ; destination name      ;[TD001]
203 000454 000533' .WORD DSACGL          ; CUG name              ;[TD001]
204 000456 000406' .WORD DSTTKL          ; object task name      ;[TD001]
205 000460 000712' .WORD RNUL           ; user name             ;[TD001]
206 000462 000733' .WORD RNPL           ; password              ;[TD001]
207 000464 000744' .WORD RNAL           ; account               ;[TD001]
208 000466 000000 .WORD 0               ; end of table          ;[TD001]
209 000470
210 000470 000560' .WORD DSCMCT          ; call mask             ;[TD001]
211 000472 000621' .WORD DSCVCT          ; call value            ;[TD001]
212 000474 000522' .WORD DSARCT          ; DTE address           ;[TD001]
213 000476 000000 .WORD 0               ; end of table          ;[TD001]
214      ;
215      ; DATA FOR DSA$DF

```

```

LNKEND - LINK BLOCK AT END OF LIST

        .SBTTL LNKEND - LINK BLOCK AT END OF LIST

        ;+
        : LNKEND - LINK BLOCK AT END OF LIST
        : INPUTS:
        :   R0 - ADDRESS OF LISTHEAD
        :   R5 - UNMAPPED ADDRESS OF CURRENT BLOCK
        : OUTPUTS:
        :   BLOCK IS LINKED AT END OF LIST
        :-
        LNKEND::GETRV R0,#2      ; READ IN LISTHEAD
10$:      MOV      R0,R1      ; SAVE ADDRESS OF PREVIOUS BLOCK
        MOV      $BFR,R0     ; GET ADDRESS OF NEXT BLOCK IN LIST
        BEQ      20$         ; BR IF END OF LIST - LINK IT HERE
        CEACCS  R0,R2      ; CONVERT TO MAPPED ADDRESS
        GETAD   R2,#2      ; READ IN LISTHEAD
        BR      10$         ; GET NEXT BLOCK IN LIST
20$:      MOV      R5,-(SP)   ; GET UNMAPPED ADDR OF CURRENT BLOCK
        MOV      R1,-(SP)   ; GET ADDRESS OF PREVIOUS BLOCK
        CALL    $XLINK      ; LINK BLOCK INTO LIST
        RETURN

```

VCFP CREATED BY MACRO ON 29-JUN-85 AT 02:18 PAGE 5 E 10
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
\$ERRO1	001146 RG	#8-263
\$ERRO2	001222 RG	#8-264
\$ERRO5	001276 RG	#8-265
\$ERRO6	001332 RG	#8-266 14-547
\$ERRO9	001352 RG	#8-267
\$ERRP7	000766 RG	#8-259
\$ERRP8	001022 RG	#8-260
\$ERRP9	001056 RG	#8-261
\$ERR1P	001602 RG	#8-273 10-431
\$ERR1Q	001640 RG	#8-274 11-461
\$ERR1R	001676 RG	#8-275 11-462
\$ERR1T	001724 RG	#8-276
\$FNOUT	= ***** GX	8-274 13-513
\$GCLEP	= ***** GX	11-446
\$GETAD	= ***** GX	16-633
\$GETRV	= ***** GX	16-628
\$NTCFP	000000 RG	#9-303
\$PDVNM	= ***** GX	17-657
\$PDVTA	= ***** GX	17-656
\$PFLAG	= ***** GX	9-312 17-668
\$QCHN	= ***** GX	9-361 9-344 9-363 9-381 19-737
\$QCHG	= ***** GX	9-367
\$QDSA	= ***** GX	9-375
\$QDSC	= ***** GX	9-379
\$QDSM	= ***** GX	9-385
\$QDS	= ***** GX	9-355
\$QDTE	= ***** GX	9-333
\$QPSN	= ***** GX	9-310
\$QPVC	= ***** GX	9-365
\$QRDT	= ***** GX	9-337
\$QRNA	= ***** GX	9-350
\$QRNW	= ***** GX	9-346
\$QY29	= ***** GX	9-339
\$QX3P	= ***** GX	9-316 9-328
\$RADDR	= ***** GX	*16-628 *16-633
\$REP.N	= ***** GX	8-273 8-274
\$TIOUT	= ***** GX	12-495 13-516
\$XLINK	= ***** GX	16-637
.PCHAR	= ***** GX	18-687 18-689
.PNUMB	= ***** GX	14-535 14-542 14-554 14-568
.PNUMH	= ***** GX	14-533 14-540 14-552 14-566
.PSTCN	= ***** GX	14-524 19-740 19-745 19-751
.PSTPT	= ***** GX	14-526 19-752

VCI0
 MACRO
 MACRO
 ASLS
 CALL
 DHBDF
 HWDDE
 LBLDF
 PUTRC
 RAND
 RETURN

VCIO CREATED BY MACRO ON 29-JUN-85 AT 02:18 PAGE 2 E 11

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
ASL\$	#5-70 8-203
CALL	7-148 7-153
DHBD\$	#5-55 5-62
HWDD\$	#5-55 5-60
LBD\$	#5-55 5-61
PUTRC	#5-55 9-251
RAND	#9-251 9-251
RETURN	7-161 8-211
	7-155 7-156 8-190 8-207 9-247 9-251 10-280 10-302
	10-302 #10-302 10-302 9-262 10-284 10-304

VCLO

SYMB

SYMB

LX.CI

LX.LI

LX.LO

LX.NO

LX.OB

LX.PR

OF\$IN

OF\$LO

OF\$OF

OF\$ON

OF\$SM

OP\$AD

OP\$AL

OP\$CC

OP\$CI

OP\$CC

OP\$CC

OP\$CC

OP\$DS

OP\$DT

OP\$DL

OP\$ES

OP\$EV

OP\$FT

OP\$GF

OP\$HC

OP\$HT

OP\$IN

OP\$IN

OP\$KL

OP\$KL

OP\$KL

OP\$KN

OP\$KN

OP\$KN

OP\$LN

OP\$LN

OP\$LN

OP\$LN

OP\$LN

OP\$LN

OP\$LN

OP\$LN

OP\$LN

OP\$LN

OP\$LN

OP\$LN

VCLORM CREATED BY MACRO ON 29-JUN-85 AT 02:18 PAGE 3 E 12

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
LX.CIR	= 000034	#5-59
LX.LIN	= 000006	#5-59
LX.LOG	= 000021	#5-59
LX.NOD	= 000015	#5-59
LX.OBJ	= 000023	#5-59
LX.PRO	= 000005	#5-59
OF\$INS	= 000001	#5-59
OF\$LOG	= 000100	#5-59
OF\$OFF	= 000002	#5-59
OF\$ON	= 000000	#5-59
OF\$SMC	= 000200	#5-59
OP\$ADD	= 000200	#5-59
OP\$ALL	= 000200	#5-59
OP\$CCS	= 000004	#5-59
OP\$CIE	= 004000	#5-59
OP\$CON	= 000400	#5-59
OP\$COP	= 000001	#5-59
OP\$COS	= 000100	#5-59
OP\$DST	= 000200	#5-59
OP\$DTE	= 000040	#5-59
OP\$DUP	= 000002	#5-59
OP\$EST	= 000100	#5-59
OP\$EVE	= 010000	#5-59
OP\$FIL	= 000200	#5-59
OP\$GRP	= 000100	#5-59
OP\$HOS	= 000001	#5-59
OP\$HTM	= 000002	#5-59
OP\$INA	= 004000	#5-59
OP\$INC	= 001000	#5-59
OP\$KDS	= 002000	#5-59
OP\$KDT	= 000400	#5-59
OP\$KEV	= 001000	#5-59
OP\$KGR	= 001000	#5-59
OP\$KNT	= 010000	#5-59
OP\$KSK	= 002000	#5-59
OP\$LCT	= 000200	#5-59
OP\$LNE	= 000040	#5-59
OP\$LOC	= 000001	#5-59
OP\$LST	= 000002	#5-59
OP\$LTM	= 000001	#5-59
OP\$LVL	= 000010	#5-59
OP\$MAC	= 000010	#5-59
OP\$MCO	= 000002	#5-59
OP\$MDE	= 000400	#5-59
OP\$MLI	= 000004	#5-59
OP\$MON	= 000100	#5-59
OP\$MST	= 000020	#5-59
OP\$NAM	= 000004	#5-59
OP\$NET	= 004000	#5-59
OP\$NLI	= 000002	#5-59
OP\$NGD	= 000020	#5-59
OP\$OUT	= 002000	#5-59

#5-59

#5-59 #5-59

VDEV
DVSYM

3

VDEV - VNP LOAD DEVICE STRUCTUR MACRO V05.03b Tuesday 03-Sep-85 10:28 Page 10
 DVSYM - DEFINE SPECIAL SYMBOLS

```

306          .SBTTL DVSYM - DEFINE SPECIAL SYMBOLS
307
308          ;+
309          DVSYM - DEFINE SPECIAL DATA STRUCTURE SYMBOLS
310
311          THIS ROUTINE IS CALLED TO SCAN THE [131,54]XXXTAB.STB FILE FOR THE FOLLOWING
312          SPECIAL SYMBOLS:
313          $XXDCB          DCB ADDRESS
314          $XXEND          END OF DEVICE TABLES
315
316          INPUTS:
317          $$DEV=2 CHARACTER DEVICE NAME
318          $NAME=PROCESS NAME
319
320          OUTPUTS:
321          C-BIT=SUCCESS/FAILURE
322          DCB=VALUE OF $XXDCB FROM STB FILE
323          LENGTH=DITTO $XXEND
324
325          DVSYM: MOV      $NAME,R0          ; SETUP DEFAULT FILE NAME
326                  MOV      #RTAB,R1        ; OF XXXTAB
327                  CALL      $DFN2
328                  MOV      $$DEV,DNAME+1    ; PREFACE DEVICE NAME WITH '$'
329                  MOV      $$DEV+1,DNAME+2
330                  MOV      #DNAME,R0        ; CONVERT IT TO RAD50
331                  MOV      PC,R1
332                  CALL      $CAT5
333                  MOV      #DBLK1,R2        ; POINT AT FIRST CONTROL BLOCK
334                  MOV      R1,4(R2)        ; SETUP SPECIAL NAME
335                  MOV      (R2),R2        ; GET NEXT CONTROL BLOCK ADDRESS
336                  BNE      10$            ; IF NE, NOT AT END
337                  MOV      R1,DUMMY        ; SETUP DISPATCH TABLE NAME
338                  MOV      FOUND,DCB      ; DO STRUCTURES ALREADY EXIST?
339                  CLC                      ; ASSUME SUCCESS
340                  BNE      60$            ; IF NE, YES
341                  CLR      R0
342                  CLR      R1
343                  MOV      #STBEXT,R2      ; SETUP ASCIZ FILE SPEC
344                  CALL      $AZFS          ; FOR [131,54]XXXTAB.STB
345                  MOV      #DBLK1,R0
346                  CALL      $SYMBL        ; POINT AT FIRST SYMBOL CONTROL BLOCK
347                  BCS      101$          ; SCAN THE SIB FILE
348                  MOV      #DBLK1,R1      ; IF CS, FILE ERROR
349                  CLR      R2
350                  TST      2(R1)          ; POINT AT FIRST CONTROL BLOCK
351                  BNE      30$            ; TALLY NUMBER OF ERRORS IN R2
352                  MOV      #-1,LENGTH    ; IS THE $XXEND SYMBOL DEFINED?
353                  MSG$      2A           ; IF NE, YES
354                  INC      R2             ; FAKE OUT LENGTH CHECK LATER
355                  MOV      (R1),R1        ; PRINT ERROR MESSAGE
356                  BEQ      50$            ; TALLY THE ERROR
357                  TST      2(R1)          ; GET NEXT CONTROL BLOCK ADDRESS
358                  BEQ      20$            ; IF EQ, END OF LIST
359                  BIT      #1,10(R1)      ; IS SYMBOL DEFINED?
360                  BEQ      40$            ; IF EQ, NO
361                  MSG$      2N           ; IS SYMBOL VALUE ODD?
362                  BR       5$            ; IF EQ, NO
363                  BR       5$            ; PRINT ERROR MESSAGE
364                  CMP      10(R1),LENGTH ; IS SYMBOL WITHIN TABLE LIMITS?

```

VDEV - VNP LOAD DEVICE STRUCTUR MACRO V05.03b Tuesday 03-Sep-85 10:28 Page 17-2
Symbol table

U.SCB = 000020	U.UNIT = 000006	U2.R04 = 100000	\$DEA16 = ***** GX	\$ERR2X = 003334R	002
U.SHST = 000076	U.UNSZ = 000114	U2.SCS = 000004	\$DEVHD = ***** GX	\$ERR2Y = 003376R	002
U.SHJN = 000074	U.UNTI = 000060	U2.SLV = 000200	\$DFN2 = ***** GX	\$FEATR = ***** GX	
U.SPC = 000036	U.USVR = 000106	U2.VT5 = 000002	\$ERR2A = 002204R	002 \$GETRV = ***** GX	
U.STS = 000005	U.UTIL = 000036	U2.7CH = 01000C	\$ERR2B = 002226R	002 \$IOSB = ***** GX	
U.ST2 = 000007	U.VCB = 000034	U3.OPA = 100000	\$ERR2C = 002264R	002 \$LBUF = ***** GX	
U.SUB = 000036	U.VSER = 000120	U3.PAR = 040000	\$ERR2D = 002322R	002 \$LLEN = ***** GX	
U.ICHP = 000042	U.2MED = 000070	U3.UPC = 020000	\$ERR2E = 002350R	002 \$NAME = ***** GX	
U.TCVP = 000043	U2.AT = 000020	U4.CR = 000100	\$ERR2F = 002376R	002 \$NLDEV = 000000RG	
U.TFLK = 000040	U2.CRT = 002000	VAD = 002014R	\$ERR2H = 002434R	002 \$PDVA = ***** GX	
U.TFRQ = 000036	U2.DH1 = 100000	VREL = 000200R	002 \$ERR2J = 002464R	002 \$PRV.N = ***** GX	
U.TIXL = 000060	U2.DJ1 = 040000	V\$CTR = 001000	\$ERR2K = 002536R	002 \$PUTRC = ***** GX	
U.TLFP = 000053	U2.DZ1 = 000100	X\$DBT = 000000	\$ERR2L = 002600R	002 \$RADDR = ***** GX	
U.TMPI = 000047	U2.ESC = 001000	Z.PCB = ***** GX	\$ERR2M = 002634R	002 \$READ = ***** GX	
U.TRCK = 000100	U2.HFF = 010000	\$ALLT5 = ***** GX	\$ERR2N = 002672R	002 \$REP.C = ***** GX	
U.TSTA = 000026	U2.HLD = 000040	\$AZFS = ***** GX	\$ERR2P = 002706R	002 \$REP.P = ***** GX	
U.TST5 = 000054	U2.LOG = 000400	\$BFR = ***** GX	\$ERR2Q = 002744R	002 \$RLBL = ***** GX	
U.TST6 = 000056	U2.LWC = 000001	\$CAT5 = ***** GX	\$ERR2R = 003000R	002 \$RSIZE = ***** GX	
U.TTAB = 000050	U2.L3S = 000004	\$CLDEQ = ***** GX	\$ERR2S = 003052R	002 \$SAVAL = ***** GX	
U.TTYP = 000046	U2.LBS = 010000	\$CLIOS = *** * GX	\$ERR2T = 003104R	002 \$SYMBL = ***** GX	
U.TUX = 000024	U2.NEC = 004000	\$CLOPE = ***** GX	\$ERR2U = 003156R	002 \$TIOUT = ***** GX	
U.UHVR = 000107	U2.PRIV = 000010	\$CUR.N = ***** GX	\$ERR2V = 003230R	002 \$\$DEV = ***** GX	
U.UIC = 000044	U2.RMT = 020000	\$CSTA = ***** GX	\$ERR2W = 003262R	002 \$\$TBL = ***** GX	
U.UNFL = 000052					

. ABS. 177776 000 (RW,I,GBL,ABS,OVR)
002372 001 (RW,I,LCL,REL,CON)
DATA 003560 002 (RW,D,LCL,REL,CON)
..BUF 000002 003 (RW,D,LCL,REL,CON)
Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 12744 Words (50 Pages)
Size of core pool: 14440 Words (55 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:45.25

DB2:VDEV.T50,[132,134]VDEV/CR/-SP=DB2:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VDEV

VDIS - VNP SHOW COMMANDS
LOCAL DATA

MACRO V05.03b Monday 15-Jul-85 12:14 ^{E 15} Page 6-3

276 001704 000000

.WORD 0

; END OF TABLE

VDIS - VNP SHOW COMMANDS

MACRO V05.03b Monday 15-Jul-85 12:14 ^{F 15} Page 7

```

799          .SBTIL DLOTYP - SHOW LOGGING SINK TYPE
800
801          ;+
802          ; DLOTYP - DISPLAY LOGGING SINK TYPE
803
804          ; INPUTS:
805          ; R5 - POINTER TO SINK TYPE TABLE IF KNOWN OR ACTIVE LOGGING
806          ; $QUAL - COMMAND QUALIFIERS
807          ; $OPTON - COMMAND OPTIONS
808
809          ; OUTPUTS:
810          ; CARRY CLEAR:
811          ; R4 - LOGGING STATE BIT
812          ; LOGGING SINK TYPE DISPLAYED ON USER'S TERMINAL
813
814          ; CARRY SET:
815          ; LOGGING SINK TYPE NOT DISPLAYED (LOGGING NOT ACTIVE)
816          ; R1,R2 DESTROYED
817
818          ;-
819
820 004600 012701 000323' DLOTYP: MOV #LOMON,R1 ; ASSUME LOGGING MONITOR
821 004604 012704 000004 MOV #FF,MON,R4 ;
822 004610 032767 000100 000000G BIT #OP$MON,$OPTON ; OPERATE ON LOGGING MONITOR?
823 004616 001014 BNE 10$ ; BR IF YES
824 004620 012701 000340' MOV #LOCON,R1 ; ASSUME LOGGING CONSOLE
825 004624 012704 000001 MOV #FF,CON,R4 ;
826 004630 032767 000400 000000G BIT #OP$CON,$OPTON ; OPERATE ON LOGGING CONSOLE?
827 004636 001004 BNE 10$ ; BR IF YES
828 004640 012701 000333' MOV #LOFIL,R1 ; ELSE MUST BE LOGGING FILE
829 004644 012704 000002 MOV #FF,FIL,R4 ;
830 004650 022767 000001 000000G 10$: CMP #QF$ACT,$QUAL ; SHOW ACTIVE LOGGING?
831 004656 001005 BNE 20$ ; BR IF NO
832 004660 030467 000000G BIT R4,$LGSTT ; IS THIS SINK TYPE ACTIVE?
833 004664 001002 BNE 20$ ; BR IF YES
834 004666 000261 SEC ; ELSE TYPE MESSAGE NOT DISPLAYED
835 004670 000411 BR 40$
836 004672 012702 001034' 20$: MOV #TEMP1,R2 ; POINT TO TEMPORARY STORAGE AREA
837 004676 112122 30$: MOVB (R1)+,(R2)+ ; STORE SINK TYPE
838 004700 001376 BNE 30$ ; LOOP TILL DONE
839 004702 ERRPTS LOMSG6,MSGOUT ; DISPLAY LOGGING SINK TYPE
840 004712 000241 CLC
841 004714 40$: RETURN

```

```

131      ; SYMBOL TABLE EXTENSION
132
133 000402      123      124      102 STBEXT: .ASCIZ  'STB'
134
135      .EVEN
136
137      ;
138      ; COMM EXEC PCB ADDRESS FOR 11M/S, OR TEMPORARY STORAGE FOR M-PLUS
139      ;
140      .IF      NDF,R$$MPL
141 000406      CEXPCB: .BLKW  1
142      .IFF
143      TMPSTR: .BLKW  1
144      .ENDC
145
146      ;
147      ; LOCAL CLOCK QUEUE BLOCK
148
149 000410      CLKQB: .BLKB  C.LGTH
150
151      ;
152      ; NTINIT RAD50 NAME
153
154 000430      055251  054374      NTINAM: .RAD50  /NTINIT/
155
156      ;
157      ; RSX11M SYMBOLS USED FOR CEX VALIDATION
158      ;
159      .IF NDF R$$MPL
160
161      CBLK1: .WORD  CBLK3,0
162      .RAD50  '$CLINS..'
163 000440      124504  035203      .WORD  0,.$CLINS
164 000444      000000  000000G      CBLK3: .WORD  CBLK4,0
165 000450      000464  000000      .RAD50  '$FRKHD'
166 000454      124702  043004      .WORD  0,.$FRKHD
167 000460      000000  000000G      CBLK4: .WORD  CBLK5,0
168 000464      000500  000000      .RAD50  '$HEADR'
169 000470      125005  003362      .WORD  0,.$HEADR
170 000474      000000  000000G      CBLK5: .WORD  XBLK1,0
171 000500      000514  000000      .RAD50  '$$TKDP..'
172 000504      125714  042560      .WORD  0,.$TKDP
173 000510      000000  000000G      .ENDC
174
175      ;
176      ; COMM EXEC SYMBOLS USED FOR NON VECTORED PROCESS VALIDATION
177      ;
178      ;
179      XBLK1: .WORD  XBLK2,0
180 000514      000526  000000      .RAD50  '$CCBRT..'
181 000520      124473  007544      .WORD  0
182 000524      000000  000000      XBLK2: .WORD  XBLK3,0
183 000526      000540  000000      .RAD50  '$CCBGT'
184 000532      124473  006654      .WORD  0
185 000536      000000  000000      XBLK3: .WORD  0,0
186 000540      000000  000000      .RAD50  '$ZTIME'
187 000544      126344  035115

```

```

707 001410 012600      MOV      (SP)+,R0      ; GET OUR ENTRY'S ADDRESS
708 001412             GETRV      R3          ; READ IN PREVIOUS ENTRY
709 001422 010014      MOV      R0,(R4)       ; POINT PREVIOUS ENTRY AT NEW ONE
710 001424             PUTRC      R0          ; REWRITE PREVIOUS ENTRY
711 001430             SWTCHO     #CLKQB       ; POINT AT BUFFERED NEW ENTRY
712 001436             PUTRC      R0          ; WRITE IT TO APPROPRIATE ADDRESS
713 001446             SWTCHO     R0          ; SET BUFFER TO DEFAULT
714 001454             40$:      RETURN
715
716             ; ERRORS
717
718 001456 012600      101$:      MOV      (SP)+,R0      ; GET NODE'S ADDRESS
719 001460 012701 000020      MOV      #C.LGTH,R1      ; AND LENGTH
720 001464             CALL      $DEA16        ; AND DEALLOCATE NODE.
721 001470             ERRTPT$   ERR5,MSGOUT      ; TASK ACTIVE
722 001500 000261      SEC              ; INDICATE ERROR
723 001502             RETURN

```

VCEX
MACRO C
MACRO N
ASL\$
CALL

CALLR
CLKDF\$
DHBDF\$
EMSG\$R
ERBLK\$
ERMSG\$
ERRPT\$
GETAD
GFTRV

NILDF\$
OPTDF\$
PCBDF\$
PKTDF\$
PUTAD
PUTRC

RAND

RESRG
RETURN

SAVRG
SOB
SWTCHI
SWTCHO
TCBDF\$
VNPBPT

VCEX CREATED BY MACRO ON 29-JUN-85 AT 02:14 PAGE 6 F 3

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
ASL\$	#5-58 7-322
CALL	7-292 7-297 7-309 7-313 7-318 7-325 7-337 8-350 8-354 8-355
	8-357 8-358 8-363 8-366 8-374 8-379 8-385 9-408 9-421 9-430
	9-446 9-448 9-453 9-455 11-581 11-585 12-636 12-643 13-670 13-685
	13-692 13-697 13-708 13-710 13-712 13-720 13-721 14-738 14-741 14-753
	14-779 14-781 14-786 14-789 14-800 14-801 14-805 14-837 14-841 14-843
	15-864 15-866 15-867
CALLR	#5-59 7-299 7-315 8-394 9-508
CLKDF\$	#5-59 5-68
DHBD\$	#5-59 5-67
EMSG\$R	#5-58 9-506
ERBLK\$	#5-59 6-94 6-96 6-98 6-100 6-102 #7-291
ERMSG\$	#5-59 6-93 6-95 6-97 6-99 6-101 #7-291
ERRPT\$	#5-59 #7-291 13-721 14-741 14-753 14-781 14-843
GETAD	#5-58
GFTRV	#5-58 7-318 7-325 8-379 12-636 12-643 13-685 13-692 13-697 13-708
	14-786 14-789
NLDF\$	#5-58 5-65
OPTDF\$	#5-59 5-66
PCBDF\$	#5-58 5-64
PKTDF\$	#5-60 5-70
PUTAD	#5-58
PUTRC	#5-58 7-337 8-350 8-363 8-385 11-581 11-585 13-710 13-712 14-800
	14-805 14-837
RAND	#7-318 7-318 #7-325 7-325 #7-337 7-337 #8-350 8-350 #8-363 8-363
	#8-379 8-379 #8-385 8-385 #11-581 11-581 #11-585 11-585 #12-636 12-636
	#12-643 12-643 #13-685 13-685 #13-692 13-692 #13-697 13-697 #13-708 13-708
	#13-710 13-710 #13-712 13-712 #14-786 14-786 #14-789 14-789 #14-800 14-800
	#14-805 14-805 #14-837 14-837
RESRG	#5-60 14-742 14-754 14-782 14-844 14-850
RETURN	7-294 8-387 9-500 10-552 11-619 12-648 13-714 13-723 14-744 14-756
	14-784 14-846 14-851 15-868
SAVRG	#5-60 14-736
SOB	8-383
SWTCHI	#5-60
SWTCHO	#5-58 11-580 11-584 11-618 13-711 13-713 14-848
TCBDF\$	#5-59 5-69
VNPBPT	#7-290 7-291

VCFG -
OPEN CC

445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500

```

445 .SBTTL OPEN CONFIG FILE
446 ;+
447 ; OPEN - OPEN THE CONFIG FILE
448 ;
449 ; INPUTS:
450 ; NONE
451 ;
452 ; OUTPUTS:
453 ; C-BIT=SUCCESS/FAILURE
454 ;
455 000730 012700 000000G OPEN: MOV #SDFUIC,R0 ; SETUP ASCII FILE SPEC
456 000734 012701 000000' MOV #CFGNAM,R1 ; FOR CETA3.MAC
457 000740 012702 000006' MOV #CFGEXT,R2 ;
458 000744 CALL $AZFS ;
459 000750 CALL $CLQUE ; OPEN THE CONFIG FILE
460 000754 105777 000000G TSTB @SCLIOS ; SUCCESS? (CLEARS C-BIT)
461 000760 100401 BMI 101$ ; IF MI, NO
462 000762 RETURN
463 ;
464 ; ERROR CONDITION
465 ;
466 101$: MSG$R 1P ; OPEN FAILURE
467 000764
  
```

853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885

VCFG -
TPARS

```

1409 003144 001345      BNE  CHERR
1410 003146 016700      MOV  UCNT,R0      ; GET UNIT COUNT
1411 003152 016701 000000G  MOV  $SLTA,R1      ; AND SLT ADDRESS
1412 003156 126100 000000G  L,UNT(R1),R0      ; IS THIS THE KEY UNIT?
1413 003162 001003      BNE  10$      ; IF NE, NO
1414 003164 016767 000000G 000000G  MOV  .PNUMB,$$PCHA      ; SAVE DLC CHARACTERISTICS
1415 003172 006300      ASL  R0      ; CONVERT TO A WORD INDEX
1416 003174 016760 000000G 000002G  MOV  .PNUMB,$$PCHA+2(R0) ; STORE PARAMETER VALUE
1417 003202      RETURN

1418
1419      ;
1420      ; SECONDARY CSR (UNT$DF)
1421      ;
1422 003204 032767 000000G 000000G  U.SCSR: BIT  #FL.LMC,$FLAGS      ; COMIOP DEVICE?
1423 003212 001440      BEQ  111$      ; IF EQ, NO
1424 003214 016701 000152'      MOV  UCNT,R1      ; GET UNIT COUNT
1425 003220 032767 000000G 000000G  BIT  #FL.KMX,$FLAGS      ; COMIOP MUX DEVICE?
1426 003226 001406      BEQ  10$      ; IF EQ, NO
1427 003230 032701 000007      BIT  #7,R1      ; IS THIS THE FIRST, NINTH, ETC.?
1428 003234 001027      BNE  111$      ; IF NE, NO
1429 003236 006201      ASR  R1      ; CONVERT TO A WORD INDEX
1430 003240 006201      ASR  R1      ; ...
1431 003242 000401      BR   20$
1432 003244 006301      10$: ASL  R1      ; CONVERT TO A WORD INDEX
1433 003246 016700 000000G 20$: MOV  ..SCSR,R0      ; ASSUME SECONDARY CSR ALREADY PRESENT
1434 003252 016702 000000G      MOV  $SLTA,R2      ; GET THE SLT ADDRESS
1435 003256 032767 000000G 000000G  BIT  #MS.SEC,$MISS      ; IS THE SECONDARY CSR MISSING ?
1436 003264 001403      BEQ  25$      ; IF EQ, NO
1437 003266      23$: CALL GETCSR      ; GET THE CSR VALUE
1438 003272 103405      BCS  101$      ; IF CS, ILLEGAL CSR VALUE
1439 003274 010061 000000G 25$: MOV  R0,$$SCSR(R1) ; STORE SECONDARY CSR VALUE
1440 003300 005267 000152'      INC  UCNT      ; INCREMENT UNIT COUNT
1441 003304      RETURN
1442 003306      101$: MSG$R 1L      ; ILLEGAL SECONDARY CSR VALUE
1443 003314      111$: MSG$R 1M      ; SECONDARY CSR NOT ALLOWED

```

VCFG

SYMBOL

SYMBOL

MS.VCT

NAME

MS\$VCT

OPEN

PF.PSV

PF.XDF

PLTDF

PL.CHN

P.EXT

P.INC

QADD

QCNT

QDDM

QFEA

QLLC

QPLT

QSLT

QSTA

QUNT

READ

RTSPC

R\$EIS

R\$MPL

R\$11D

SCNT

SETMO

SLTDF

SPACE

SPEEDT

SPFILL

SPSAV

STADF

STORID

ST.APR

ST.CST

ST.HTM

ST.NUM

SYNERR

SYSFDE

S\$BAS

VCFG

VCFP

```

216      :
217      :      DARMLN == 15.      : MAX NUMBER OF DIGITS IN REMOTE DTE ADDRESS
218 000500      DSARDT::.BLKB DARMLN      : TEMP STORAGE FOR UNPACKED REMOTE DTE ADDRESS
219 000517      :      .BYTE 0      : END OF STRING INDICATOR FOR REMOTE DTE ADDR
220      :      DARDEN == -2      : END OF REMOTE DTE ADDRESS
221 000520      DSALCT::.BLKB 1      : COUNT OF DIGITS IN LOW SUBADDRESS      :**-2
222 000521      DSAHCT::.BLKB 1      : COUNT OF DIGITS IN HIGH SUBADDRESS
223 000522      DSARCT::.BLKB 1      : COUNT OF DIGITS IN REMOTE DTE ADDRESS
224 000523      DSARDP::.BLKB <DARMLN/2>+1 : PACKED DTE ADDRESS (must follow DSARCT) :[TD001]
225 000533      DSACGL::.BLKB 1      : Length of CUG name :[TD001]
226 000534      DSACUG::.BLKB MAXID      : Closed user group name :[TD001]
227      :      .EVEN
228 000554      DSASLO::.BLKW 1      : LOW END OF SUBADDRESS RANGE
229 000556      DSASHI::.BLKW 1      : HIGH END OF SUBADDRESS RANGE
230      :
231      : DATA FOR DSC$DF MACRO
232      :
233 000560      DSCMCT::.BLKB 1      : NUMBER OF HEX DIGITS IN CALL DATA MASK
234 000561      DSCMSK::.BLKB 32.      : CALL DATA MASK (must follow DSCMCT) :[TD001]
235 000621      DSCVCT::.BLKB 1      : NUMBER OF HEX DIGITS IN CALL DATA VALUE :[TD001]
236 000622      DSCVAL::.BLKB 32.      : CALL DATA VALUE (must follow DSCVCT) :[TD001]
237      :
238      : Data for RNW$DF, RNA$DF and DSN$DF macros :[TD001]
239      :
240      : *** Data fields must immediately follow length fields. *** :[TD001]
241      :
242 000662      RNTL::.BLKB 1      : Network name length :[TD001]
243 000663      RNETWK::.BLKB MAXID      : Network name :[TD001]
244 000703      RNNL::.BLKB 1      : Remote node length :[TD001]
245 000704      RNNODE::.BLKB MXNODID      : Remote node name :[TD001]
246 000705      RNUL::.BLKB 1      : User name length :[TD001]
247 000706      RNUSER::.BLKB MXACUS      : User name :[TD001]
248 000707      RNPL::.BLKB 1      : Password length :[TD001]
249 000734      RNPASS::.BLKB MXACPA      : Password :[TD001]
250 000744      RNAL::.BLKB 1      : Account length :[TD001]
251 000745      RNACCT::.BLKB MXACAC      : Account :[TD001]
252      :      .EVEN      :**-3

```

```

640 .SBTTL FNDPDV - FIND PDV
641
642 ;+
643 ; FNDPDV - TRY TO FIND PDV
644 ;
645 ; INPUTS:
646 ; R0=RAD50 PDV NAME
647 ;
648 ; OUTPUTS:
649 ; C-BIT=SUCCESS/FAILURE
650 ; R0=PDV ADDRESS
651 ; R1=PDV INDEX
652 ; R2=DESTROYED
653 ;
654 FNDPDV::
655 MOV R0, -(SP) ; SAVE NAME TO BE MATCHED
656 MOV $PDVTA, R1 ; GET PDV TABLE ADDRESS
657 MOV $PDVNM, R2 ; AND NUMBER OF PDV'S
658 BEQ 30$ ; IF ZERO, NO PDV'S!
659 MOV (R1)+, R0 ; GET PDV ADDRESS
660 BEQ 20$ ; IF ZERO, SKIP THIS ONE
661 CMP Z, NAM(R0), (SP) ; DOES THE NAME MATCH?
662 BEQ 40$ ; IF EQ, YES (C-BIT CLEAR)
663 SOB R2, 10$ ; LOOP $PDVNM TIMES
664 SEC ; INDICATE FAILURE
665 INC (SP)+ ; PURGE STACK
666 ROR (SP) ; SAVE C-BIT
667 TST ~(R1) ; CALCULATE PDV INDEX
668 SUB $PDVTA, R1 ;
669 ASL (SP) ; RESTORE C-BIT
670 RETURN

```

★ ★ F I L E

ASL\$	#5-63	15-604								
CALL	9-304	9-307	9-308	9-310	9-314	9-316	9-321	9-328	9-333	9-335
	9-337	9-339	9-346	9-348	9-350	9-352	9-359	9-361	9-365	9-367
	9-373	9-375	9-377	9-379	9-383	9-385	9-387	9-396	10-422	10-423
	11-446	12-477	12-482	12-495	12-498	13-513	14-527	16-628	16-632	16-633
	16-637	18-692	19-734	19-753						
	13-516									
CALLR	#5-64	16-632								
CEACCS	#5-63	9-403	9-404	9-405	9-406	9-407	10-431	11-461	11-462	14-547
EMSGSR	#5-64	16-633								
GETAD	#5-64	16-628								
GETRV	#5-63									
ISTATS	#5-63									
NTLR\$	#5-63	8-259	8-260	8-261	8-262	8-263	8-264	8-265	8-266	8-267
	8-268	8-269	8-270	8-271	8-272	8-273	8-274	8-275	8-276	
	#16-628	16-628	#16-633	16-633						
RAND	#5-75									
REJ\$	#5-64									
RESRG	9-398	10-426	11-451	12-501	14-529	14-536	14-543	14-562	14-575	15-612
RETURN	16-638	17-670	18-696	19-755	19-758	20-781	21-803			
	#5-64									
SAVRG	#5-64	5-67								
SLTDF\$	#5-64									
SOB	17-663									
STATE\$	#5-63									
TRAN\$	#5-63									
VNPDBG	#5-65	9-303								

VV		
VV		
VV		
VV		
VV		
VV		
VV		
VV		
VV	VV	
VV	VV	
	V	
	V	
LL		
LL		
LL		
LL		
LL		
LL		
LL		
LL		
LL		
LL	LLL	
LL	LLL	
LL	LLL	

F 11

```

LL          SSSSSSSS  TTTTTTTTTT
LL          SSSSSSSS  TTTTTTTTTT
      SS
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SSSSSS    TT
LL          SSSSSS    TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LLLLLLLLLL  SSSSSSSS  TT
LLLLLLLLLL  SSSSSSSS  TT

```

OP\$OW
OP\$PA
OP\$PR
OP\$RE
OP\$RP
OP\$SC
OP\$SE
OP\$SE
OP\$SI
OP\$ST
OP\$TF
OP\$TR
OP\$UC
OP\$US
OP\$VE
OP\$VE
OP\$XA
OP\$XK
OP\$X9
OP\$X5
QF\$AC
QF\$AL
QF\$LC
QF\$SK
ROB
SYSFE
TY\$CH
TY\$EV
TY\$ST
TY\$SU
WC .CM
WC .EV
WC .TR
WC .UN
Z .DSK
Z .NAR
\$BFR
\$CER
\$DEA
\$GE TH
\$PDVI
\$PDV
\$PUT
\$RADT
CLKM
MGMM
PDV

CREF 04.00

3
 4
 5
 6
 7
 8
 9
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 40
 41
 42
 43
 44
 45
 46
 47
 48
 49
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
 100
 101
 102
 103
 104
 105
 106
 107
 108
 109
 110
 111
 112
 113
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154
 155
 156
 157
 158
 159
 160
 161
 162
 163
 164
 165
 166
 167
 168
 169
 170
 171
 172
 173
 174
 175
 176
 177
 178
 179
 180
 181
 182
 183
 184
 185
 186
 187
 188
 189
 190
 191
 192
 193
 194
 195
 196
 197
 198
 199
 200
 201
 202
 203
 204
 205
 206
 207
 208
 209
 210
 211
 212
 213
 214
 215
 216
 217
 218
 219
 220
 221
 222
 223
 224
 225
 226
 227
 228
 229
 230
 231
 232
 233
 234
 235
 236
 237
 238
 239
 240
 241
 242
 243
 244
 245
 246
 247
 248
 249
 250
 251
 252
 253
 254
 255
 256
 257
 258
 259
 260
 261
 262
 263
 264
 265
 266
 267
 268
 269
 270
 271
 272
 273
 274
 275
 276
 277
 278
 279
 280
 281
 282
 283
 284
 285
 286
 287
 288
 289
 290
 291
 292
 293
 294
 295
 296
 297
 298
 299
 300
 301
 302
 303
 304
 305
 306
 307
 308
 309
 310
 311
 312
 313
 314
 315
 316
 317
 318
 319
 320
 321
 322
 323
 324
 325
 326
 327
 328
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 368
 369
 370
 371
 372
 373
 374
 375
 376
 377
 378
 379
 380
 381
 382
 383
 384
 385
 386
 387
 388
 389
 390
 391
 392
 393
 394
 395
 396
 397
 398
 399
 400
 401
 402
 403
 404
 405
 406
 407
 408
 409
 410
 411
 412
 413
 414
 415
 416
 417
 418
 419
 420
 421
 422
 423
 424
 425
 426
 427
 428
 429
 430
 431
 432
 433
 434
 435
 436
 437
 438
 439
 440
 441
 442
 443
 444
 445
 446
 447
 448
 449
 450
 451
 452
 453
 454
 455
 456
 457
 458
 459
 460
 461
 462
 463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479
 480
 481
 482
 483
 484
 485
 486
 487
 488
 489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525
 526
 527

DVSYM - DEFINE SPECIAL SYMBOLS

```

363 000606 103757          BLO      30$      ; IF LO, YES
364 000610          EMSG$    2P      ; PRINT ERROR MESSAGE
365 000616 000752          BR      25$      ; SET C-BIT IF ANY ERRORS OCCURRED
366 000620 005402          50$:    NEG      R2
367 000622          60$:    RETURN
368
369          ; ERROR CONDITIONS
370
371
372 000624 016000 000154' 101$:    MOV     ETAB-2(R0),R0 ; GET ERROR MESSAGE BLOCK ADDRESS
373 000630          CALLR    a(R0)+    ; PRINT ERROR MESSAGE

```

VDEV CREATED BY MACRO ON 3-SEP-85 AT 10:28 PAGE 1 F 14

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
BASE	000170 R	#6-123 *11-406 11-407 12-452 13-625
DBLK1	000126 R	#6-95 10-332 10-344 10-347
DBLK2	000140 R	6-95 #6-98 12-453
DCB	000150 R	#6-100 8-225 *10-337 13-498
DEVBUF	000204 R	#6-142 8-209 8-210 *8-227 *8-229 8-230 8-231 *8-234 *9-278
		9-279 9-282 9-288 9-291 9-293 12-445 12-446 13-490 13-564
		13-581 *13-583 13-585 *13-597 *13-601 15-691 15-692
DNAME	000004 R	#6-81 *10-327 *10-328 10-329
DSKBUF	000000 R	#7-184 11-399
DUMMY	000152 R	#6-105 8-261 *10-336
DVALL	000772 R	8-219 #12-437
DVDEA	002142 R	7-158 7-161 7-162 7-163 7-164 7-165 7-166 7-167 #15-685
DVERR	002034 R	7-148 7-149 7-150 7-151 7-152 7-153 7-154 7-155 7-156
		7-157 7-158 7-159 7-160 7-161 7-162 7-163 7-164 7-165
		7-166 7-167
DVLBL	000632 R	8-217 #11-390
DVSYS	000372 R	8-213 #10-324
DVXFR	001116 R	8-221 #13-488
D.DSP	= ***** GX	*8-227 9-288
D.NAM	= ***** GX	9-282 *13-500
D.PCB	= ***** GX	8-226 *8-229 9-281 13-580
D.UCB	= ***** GX	8-230 13-512 *13-522
D.UCBL	= ***** GX	8-231 13-557
D.UNIT	= ***** GX	9-291 9-293 13-502 13-504
ERRBUF	000017 R	#6-88 14-632
ERRMSG	000007 R	#6-86 14-664
ETAB	000156 R	#6-110 10-372
FIND	000242 R	8-211 #9-277
FMT4	003461 R	7-149 7-150 7-151 7-152 #7-174
FMT4A	003420 R	7-148 7-159 7-160 7-169 #7-173
FMT6	003511 R	7-153 7-154 7-155 7-156 7-158 #7-175
FMT9	003532 R	7-157 7-161 7-162 7-163 7-164 7-165 7-166 7-167 7-168
		#7-176
FM.4	= 000000	#7-149 #7-150 #7-151 #7-152
FM.4A	= 000000	#7-148 #7-159 #7-160 #7-169
FM.6	= 000000	#7-153 #7-154 #7-155 #7-156 #7-158
FM.9	= 000000	#7-157 #7-161 #7-162 #7-163 #7-164 #7-165 #7-166 #7-167 #7-168
FOUND	000166 R	#6-118 8-215 *9-286 10-337
F.NRBD	= ***** GX	*8-209 *8-254 *12-441 *12-445 *13-575 *13-577 *15-689 *15-692
F.RSIZ	= ***** GX	*8-226 *8-233 *8-250 *9-281
F.URBD	= ***** GX	*8-210 *8-255 *12-442 *12-446 *15-688 *15-691
LENGTH	000136 R	#6-97 *10-351 10-362 *11-407 12-438
LSBFLG	= ***** GX	11-402
LSBLDZ	= ***** GX	11-404
LSBMXZ	= ***** GX	11-404
LSBSA	= ***** GX	11-406
PBUF	000174 R	#6-133 *12-447 13-495 13-563 13-569 *13-573 *13-578 13-593 13-597
		13-600 13-631
PLEN	000176 R	#6-134 *12-448 13-489 13-562 13-565 13-566 13-570 *13-572 *13-579
REP4A	002214 R	7-148 7-159 7-160 #16-708
REP4A1	002224 R	7-169 #16-711
REP4A2	002264 R	16-710 #16-723

```

278          .SBTTL LOCAL SYMBOLS
279
280          ;+
281          LOCAL SYMBOLS
282          ; -
283
284
285
286          ;
287          LENGTH OF UCB
288          ;
289          000042          U.LEN = 34.          ; LENGTH OF UCB
290
291          ;
292          LENGTH OF DCB
293          ;
294          000036          D.LEN = 30.          ; LENGTH OF DCB
295
296          ;
297          ASCII CHARACTERS
298          ;
299          000040          SPACE = 40          ; ASCII SPACE " "
300          000054          COMMA = 54          ; ASCII COMMA ","
301          000055          DASH = 55          ; ASCII DASH "-"
302          000056          PERIOD = 56         ; ASCII PERIOD "."
303          000057          SLASH = 57          ; ASCII SLASH "/"
304
305          000100          MSKLEN = 64.         ; NUMBER OF BITS IN EVENT MASKS
306          000010          MXDTEA = 8.         ; MAXIMUM NUMBER OF BYTES IN DTE ADDR
307          000044          RCVLNG = 36.         ; LENGTH OF PACKET TO QUE TO NTINIT
308          000040          MXCALL = 32.         ; MAX BYTES IN DESTN. CALL MASK/DATA ;[TDO01]
309          000050          MXACCT = 40.         ; MAX BYTES IN USER+PASSWORD+ACCOUNT ;[TDO01]
310          000020          MAXID = 16.         ; MAX LENGTH OF NET. MGMT. 'ID-STRING' ;[TDO01]
311
312          ;
313          OFFSETS INTO TEMPORARY STORAGE AREA
314
315          001034'          EVTMSK = TEMP1      ; EVENT MASKS
316          001044'          GCLASS = EVTMSK+10 ; GLOBAL EVENT CLASS
317          001046'          GEVMSK = GCLASS+2  ; GLOBAL EVENT MASKS FOR GLOBAL CLASS
318          001204'          BITNUM = TEMP3+2   ; EVENT MASK BIT NUMBER BEING PROCESSED
319          001205'          PREVNT = BITNUM+1   ; LAST EVENT NUMBER DISPLAYED
320
321          ;
322          FLAGS BIT DEFINITION
323
324          000001          F.STRT = 1          ; FORMATTING HAS STARTED
325          000002          F.PREV = 2          ; PREVIOUS BIT IN MASK WAS SET
326          000004          F.FND = 4          ; COMPONENT HAS BEEN FOUND

```

```

843 .SBTTL DLOSTA - DISPLAY LOGGING STATE
844
845 ;+
846 ; DLOSTA - DISPLAY LOGGING STATE
847
848 ; INPUTS:
849 ; R4 - LOGGING STATE BIT
850
851 ; OUTPUTS:
852 ; LOGGING STATE MESSAGE DISPLAYED ON USER'S TERMINAL
853
854 ; R2,R3 DESTROYED
855
856 ;
857 004716 012703 001034' DLOSTA: MOV #TEMP1,R3 ; GET STORAGE FOR STATE
858 004722 012702 000064' MOV #ON,R2 ; ASSUME STATE IS ON
859 004726 030467 000000G BIT R4,$LGSTT ; IS LOGGING ON?
860 004732 001002 BNE 10$ ; BR IF YES
861 004734 012702 000077' MOV #OFF,R2 ; ELSE STATE IS OFF
862 004740 112223 10$: MOVB (R2)+,(R3)+ ; STORE STATE
863 004742 001376 BNE 10$ ;
864 004744 ERRT$ LMSG2,MSGOUT ; DISPLAY STATE MESSAGE
865 004754 RETURN ;

```

```
188 000550 000000 .WORD 0
189
190
191 ; COMM EXEC SYMBOLS USED FOR VECTORED PROCESS VALIDATION
192
193 000552 000564 000000 VXBLK1: .WORD VXBLK2,0
194 000556 124473 007544 .RAD50 '$CCBRT'
195 000562 000000 .WORD 0
196 000564 000576 000000 VXBLK2: .WORD VXBLK3,0
197 000570 124473 006654 .RAD50 '$CCBGT'
198 000574 000000 .WORD 0
199 000576 000000 000000 VXBLK3: .WORD 0,0
200 000602 126344 035115 .RAD50 '$ZTIME'
201 000606 000000 .WORD 0
202
203 ; COMM EXEC SYMBOLS USED TO INITIALIZE VARIABLES
204
205
206 000610 000622 000000 IBLK1: .WORD IBLK2,0
207 000614 124520 023547 .RAD50 '$CXFLG'
208 000620 000000 .WORD 0
209 000622 000000 000000 IBLK2: .WORD 0,0
210 000626 125721 074375 .RAD50 '$SYSNM'
211 000632 000000 .WORD 0
212
213 ; CETAB SYMBOLS TO INITIALIZE THE COMM EXEC
214
215
216 000634 000646 000000 .PDVTB: .WORD .SLTMB,0
217 000640 125504 106242 .RAD50 '$PDVTB'
218 000644 000000 .WORD 0
219 000646 000660 000000 .SLTMB: .WORD .LLCTB,0
220 000652 125704 077412 .RAD50 '$SLTMB'
221 000656 000000 .WORD 0
222 000660 000672 000000 .LLCTB: .WORD .SLTTOT,0
223 000664 125254 012742 .RAD50 '$LLCTB'
224 000670 000000 .WORD 0
225 000672 000704 000000 SLTTOT: .WORD .PDVTOT,0
226 000676 074264 077554 .RAD50 '$SLTTOT'
227 000702 000000 .WORD 0
228 000704 000000 000000 PDVTOT: .WORD 0,0
229 000710 062266 077554 .RAD50 '$PDVTOT'
230 000714 000000 .WORD 0
231
232 ; SYMBOL TABLE ERRORS
233
234
235 000716 000000G ETAB1: .WORD $ERR44 ; OPEN FAILURE
236 000720 000000G .WORD $ERR45 ; READ FAILURE
237 000722 000000G .WORD $ERR46 ; READ OVERFLOW
238 000724 000000G .WORD $ERR47 ; BAD STB FORMAT
239
240 ; PROCESS I/O ERRORS
241
242
243 000726 000000G ETAB2: .WORD $ERR51 ; OPEN FAILURE
244 000730 000101 .WORD 101 ; GET ATTRIBUTES FAILED
```

[illegible]


```

469          .SBTIL READ - READ NEXT RECORD
470          ;+
471          ; READ - READ NEXT RECORD FROM CONFIG FILE
472          ;
473          ; INPUTS:
474          ;     NONE
475          ;
476          ; OUTPUTS:
477          ;     C-BIT=SUCCESS/FAILURE
478          ;     CFGSZ=RECORD LENGTH
479          ;
480 000772 012767 002472' 000000G READ:  MOV    #CFGBF,$CLBUF    ; SET GCL BUFFER ADDRESS
481 001000 012767 000204 000000G      MOV    #132,$CLSZ      ; AND BUFFER SIZE
482 001006      CALL    $GCLBP      ; READ NEXT RECORD
483 001012 016700 000000G      MOV    $CLIOS,R0      ; GET GCL IOSB ADDRESS
484 001016 016067 000002 002700'      MOV    2(R0),CFGSZ    ; SET RECORD LENGTH
485 001024 105710      TSTB    (R0)      ; SUCCESS? (CLEARS C-BIT)
486 001026 100401      BMI     101$      ; IF MI, NO
487 001030      10$:    RETURN
488
489          ;
490          ; ERROR CONDITIONS
491          ;
492 001032 122710 000000G      101$:  CMPB    #IE.EOF,(R0)    ; IS IT END OF FILE?
493 001036 000261      SEC      ; ASSUME THAT IT IS
494 001040 001773      BEQ     10$      ; IF EQ, YES - RETURN WITH C-BIT SET
495 001042 122710 000000G      CMPB    #IE.RBG,(R0)    ; IS IT BUFFER TOO SMALL?
496 001046 001403      BEQ     111$      ; IF EQ, YES
497 001050      EMMSG$R  10      ; READ FAILURE
498 001056      111$:  EMMSG$R  1R      ; READ OVERFLOW

```

887
 888
 889
 890
 891
 892
 893
 894
 895
 896
 897
 898
 899
 900
 901
 902
 903
 904
 905
 906
 907
 908
 909
 910
 911
 912
 913
 914
 915
 916
 917
 918
 919
 920
 921
 922
 923
 924
 925
 926

VCFG -
TPARS

```

1445
1446      ; SETTMO - SET UP TIMEOUT VALUE
1447
1448      ; INPUTS:
1449      ; RO - UNIT NUMBER
1450
1451      SETTMO: SAVRG <RO,R1>      ; SAVE SOME REGISTERS
1452      MOV      .PNUMB,-(SP)      ; GET CHARACTERISTICS WORD 1
1453      BIC      #*C<7>,(SP)      ; ISOLATE PROTOCOL BITS
1454      CMP      #C1.DCP,(SP)+    ; IS THIS A DDCMP LINE?
1455      BNE      40$              ; BR IF NO
1456      CMPB     L.UNT(R1),RO      ; IS THIS THE KEY UNIT?
1457      BNE      5$              ; BR IF NO
1458      MOV      $$DCHA,-(SP)      ; GET CHARACTERISTICS WORD 0
1459      BR       7$              ;
1460      ASL      RO              ; CONVERT TO DOUBLE WORD INDEX
1461      ASL      RO              ;
1462      MOV      $$DCHA+4(RO),-(SP) ; GET CHARACTERISTICS WORD 0
1463      SWAB     (SP)            ; GET HIGH BYTE
1464      BIC      #*C<17>,(SP)      ; ISOLATE SPEED BITS
1465      MOV      (SP),R2          ; SAVE THEM
1466      ASL      (SP)            ; CONVERT TO WORD INDEX
1467      ADD      #SPEEDT,(SP)      ; INDEX INTO TRANSLATION TABLE
1468      MOV      @((SP)+,RO)      ; GET CORRECT SPEED
1469      MOV      $SLTA,R1         ; GET SLT ADDRESS
1470      CMPB     L.DLC(R1),L.DDM(R1) ; IS THIS A COMBINED DDM/DLC?
1471      BEQ      20$            ; BR IF YES
1472      MOV      #8,R1           ; ASSUME 8 BITS
1473      BIT      #CH.SYN,$$DCHA    ; IS TYPE SYNCH?
1474      BEQ      10$            ; BR IF YES
1475      MOV      #10,R1          ; ELSE MUST BY ASYNCH
1476      CALL     $DIV            ; CALCULATE SPEED/# BITS
1477      MOV      RO,R1           ; GET RESULT
1478      MOV      $RDBSZ,RO        ; GET SIZE OF RDB'S
1479      CALL     $DIV            ; CALCULATE TIME OUT VALUE
1480      ADD      #2,RO           ; ADD FUDGE FACTOR
1481      BIC      #*C<37>,RO       ; ISOLATE TIMEOUT VALUE
1482      SWAB     RO             ; GET IN HIGH BYTE
1483      BR       50$            ; AND FINISH UP
1484
1485      ; DDM = DLC
1486
1487      20$: MOV      #2,RO        ; GET NUMBER OF BUFFERS (ASSUME SPEED <=9600)
1488      CMP      R2,#17          ; EXTERNAL CLOCK?
1489      BNE      30$            ; BR IF NO
1490      ASL      RO             ; ELSE NUMBER OF BUFFERS = 4
1491      SWAB     RO             ; GET INTO HIGH BYTE
1492      BR       50$            ; AND STORE NUMBER OF BUFFERS
1493
1494      ; NON-DDCMP LINE
1495
1496      40$: CLR      RO          ;
1497
1498      50$: CLRB     .PNUMB+1      ; STORE TIMEOUT VALUE (OR NUMBER OF BUFFERS)
1499      BIS      RO .PNUMB        ;
1500      RESRG    <R1,RO>         ; RESTORE REGISTERS
1501      RETURN

```

VCFG

SYMBOL

SYMBOL

S.BRA
S.COST
S.CTIM
S.CTL
S.HTIM
S.NAME
S.PRI
S.UNT
UCNT

UMRFL

UNT

UNTDF

U.CHA0

U.CHA1

U.CST

U.DPR

U.HTIM

U.LTIM

U.PECH

U.SCSR

U.UNT

U.XCSR

WINDSMX

Z.NAM

\$ALPHA

\$ANY

\$AZFS

\$BFR

\$BLANK

\$CEACC

\$CLBUF

\$CLDEQ

\$CLFLG

\$CLIOS

\$CLQUE

\$CLSAV

\$CLSBK

\$CLSIZ

\$CTIM

\$CSTA

\$DFUIC

\$DIGIT

\$DIV

\$DNUME

\$EOS

\$ERRYA

\$ERRYE

\$ERRYC

\$ERRYC

VCFG CREATED BY MACRO ON 15-JUL-85 AT 19:06 PAGE 4 G 7

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
S.BRA	001752 R	7-216
S.COST	= ***** GX	7-216 7-217 7-217 7-218 7-218
S.CTIM	002014 R	#25-1146
S.CTL	001706 R	*25-1243
S.HTIM	002040 R	#25-1165
S.NAME	001512 R	#25-1129
S.PRI	001762 R	#25-1174
S.UNT	001730 R	#25-1073
UCNT	000152 R	#25-1153
		#7-153
		30-1510
UMRFL	000154 R	*8-265
UNT	000016 R	*30-1512
UNTDF	000514 R	8-338
U.CHAO	002762 R	28-1360
U.CHAT	003066 R	28-1371
U.CST	003602 R	28-1394
U.DPR	003634 R	28-1410
U.HTIM	003644 R	28-1424
U.LTIM	003670 R	*28-1440
U.PECH	003140 R	
U.SCSR	003204 R	
U.UNT	002716 R	
U.XCSR	003536 R	
WNSMX	= 000177	
Z.NAM	= ***** GX	
\$ALPHA	= 000022	
\$ANY	= 000020	
\$AZFS	= ***** GX	
\$BFR	= ***** GX	
\$BLANK	= 000006	
\$CEACC	= ***** GX	
\$CLBUF	= ***** GX	
\$CLDEQ	= ***** GX	
\$CLFLG	= ***** GX	
\$CLIOS	= ***** GX	
\$CLQUE	= ***** GX	
\$CLSAV	= ***** GX	
\$CLSBK	= ***** GX	
\$CLSIZ	= ***** GX	
\$CTIM	= ***** GX	
\$CSTA	= ***** GX	
\$DFUIC	= ***** GX	
\$DIGIT	= 000024	
\$DIV	= ***** GX	
\$DNUMB	= 000014	
\$EOS	= 000012	
\$ERRYA	001702 R	
\$ERRYB	001734 R	
\$ERRYP	001770 R	
\$ERRYQ	002012 R	
\$ERRYR	002052 R	

VCFP
ERRO

```

254          .SBTTL  ERROR MESSAGES
255          ;
256          ; ERROR MESSAGES
257          ;
258          ;***-1
259 000766      NTLERS$ <:,P7,10,CERR,RTSPC,CFLIN,<default block size>
260 001022      NTLERS$ <:,P8,10,CERR,RTSPC,CFLIN,<maximum block size>
261 001056      NTLERS$ <:,P9,10,CERR,RTSPC,CFLIN,<default window size>
262 001112      NTLERS$ <:,00,10,CERR,RTSPC,CFLIN,<maximum window size>
263 001146      NTLERS$ <:,01,8,CERR,RTSPC,CFLIN,<Default block size exceeds maximum>
264 001222      NTLERS$ <:,02,8,CERR,RTSPC,CFLIN,<Default window size exceeds maximum>
265 001276      NTLERS$ <:,05,10,CERR,RTSPC,CFLIN,<counter timer value>
266 001332      NTLERS$ <:,06,10,CERR,RTSPC,CFLIN,<line-id>
267 001352      NTLERS$ <:,09,10,CERR,RTSPC,CFLIN,<logical channel number>
268 001412      NTLERS$ <:,M4,8,CERR,RTSPC,,<X3P$DF missing>
269 001442      NTLERS$ <:,M5,8,CERR,RTSPC,,<DSA$DF missing>
270 001472      NTLERS$ <:,M6,8,CERR,RTSPC,,<DSC$DF missing>
271 001522      NTLERS$ <:,MA,8,CERR,RTSPC,,<DSN$DF missing>
272 001552      NTLERS$ <:,MB,8,CERR,RTSPC,,<RNA$DF missing>
273 001602      NTLERS$ <:,1P,8,CFERR,$REP.N,$FNOUT,<Open Failure (-***.)>
274 001640      NTLERS$ <:,1Q,8,CFERR,$REP.N,$FNOUT,<Read Failure (-***.)>
275 001676      NTLERS$ <:,1R,8,CFERR,RTSPC,FNLIN,<Read Overflow>
276 001724      NTLERS$ <:,1T,8,CERR,RTSPC,CFLIN,<Syntax Error>
277          ;***-1
278          ;
279          ; ERROR MESSAGE FORMAT STRINGS
280          ;
281          .NLIST  BEX
282 001752      103      157      156  FMT8:: .ASCIZ  "Config File -- "
283 001772      103      157      156  FMT10:: .ASCIZ  "Config File -- Illegal "
284          .EVEN
285          .LIST  BEX
286          ;***-1
287 000000      .PSECT
  
```

```

672                                     .SBTTL STRNXT - STORE NEXT DIGIT
673
674                                     ;+
675                                     : STRNXT - STORE NEXT DIGIT
676                                     :
677                                     : INPUTS:
678                                     : .PCHAR - MATCHED CHARACTER FROM TPARS TRANSITION
679                                     : NEXT - ADDRESS OF NEXT BYTE AVAILABLE FOR STORAGE
680                                     :
681                                     : OUTPUTS:
682                                     : NEXT - ADDRESS OF NEXT AVAILABLE BYTE
683                                     : COUNT - UPDATED
684                                     : R0,R1 DESTROYED
685                                     :
686                                     :
687 001254 116767 000000G 000357' STRNXT::MOVB .PCHAR,CVTBUF ; MOVE DIGIT TO CONVERSION BUFFER
688 001262 012701 000017          MOV #17,R1 ; ASSUME CHAR IS AN '*'
689 001266 126727 000000G 000052 CMPB .PCHAR,#'* ; IS CHAR AN '*' ?
690 001274 001404          BEQ 10$ ; BR IF YES
691 001276 012700 000357' MOV #CVTBUF,R0 ; POINT TO CONVERSION BUFFER
692 001302          CALL $CDTB ; CONVERT TO BINARY
693 001306 110177 000352' 10$: MOVB R1,@NEXT ; STORE BINARY DIGIT
694 001312 005267 000352' INC NEXT ; POINT TO NEXT AVAILABLE BYTE
695 001316 105267 000356' INCB COUNT ; INCREMENT DIGIT COUNT
696 001322          RETURN

```

[illegible]

```

LL          SSSSSSSS  TTTT TTTT TTT
LL          SSSSSSSS  TTTT TTTT TTT
          SS          TT
LL          SS         TT
LL          SS         TT
LL          SS         TT
LL          SS         TT
LL          SSSSSS    TT
LL          SSSSSS    TT
LL          SSSSSS    TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LLLLLLLLLLL SSSSSSSS  TT
LLLLLLLLLLL SSSSSSSS  TT

```

VCLQRM - REMOVE ENTRIES FROM CL MACRO V05.03b Saturday 29-Jun-85 6 11
Table of contents

5-	54	MACRO DEFINITIONS
6-	61	LOCAL DATA
7-	69	CLOREM - REMOVE ENTRIES FROM CLOCK QUEUE
8-	141	FNDPDV - FIND PDV ADDRESS

VCLQRM
MACRO
MACRO
CALL
CLKDF
GETRV
PUTRC
RAND
RETUR
VNPDE

VCLQRM CREATED BY MACRO ON 29-JUN-85 AT 02:18 PAGE 5 G 12

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME REFERENCES

CALL	7-89	7-94	7-101	7-127	7-129	7-134		
CLKDF\$	#5-56	5-58						
GETRV	#5-56	7-94	7-101	7-127				
PUTRC	#5-56	7-129						
RAND	#7-94	7-94	#7-101	7-101	#7-127	7-127	#7-129	7-129
RETURN	7-139	8-173						
VNPDF\$	#5-56	5-59						

VDEV
DBLBL

VDEV - VNP LOAD DEVICE STRUCTUR MACRO V05.03b Tuesday 03-Sep-85 10:28 Page 11
 DBLBL - READ LABEL BLOCK

```

375          .SBTTL DBLBL - READ LABEL BLOCK
376
377      ;+
378      ; DVLBL - OPEN IMAGE FILE AND READ LABEL BLOCK
379      ; THIS ROUTINE IS CALLED TO OPEN THE DATA STRUCTURE FILE AND TO READ THE LABEL
380      ; BLOCK. THE SYMBOL VALUES FROM THE STB FILE ARE ADJUSTED BY SUBTRACTING THE
381      ; BASE ADDRESS OF THE DATA STRUCTURE IMAGE FILE.
382
383      INPUTS:
384          DTNAM=FILE NAME
385
386      OUTPUTS:
387          C-BIT=SUCCESS/FAILURE
388          RO=LABEL BLOCK BUFFER ADDRESS
389
390      DVLBL: CLR      RO          ; SETUP ASCII FILE SPEC
391             CLR      R1          ; FOR [131,54]XXXTAB.TSK
392             CLR      R2          ; ...
393             CALL     $AZFS       ; ...
394             CALL     $CLOPE      ; OPEN THE IMAGE FILE
395             TSTB     @%CLIOS     ; SUCCESS?
396             BMI      101$        ; IF M1, NO
397             BIS      (RO)+,(RO)  ; IS FILE CONTIGUOUS?
398             BEQ      121$        ; IF EQ, NO
399             MOV      #DSKBUF,RO  ; GET ADDRESS OF DISK BLOCK SIZED BUFFER
400             CALL     $RLBL       ; READ LABEL BLOCK
401             BCS      131$        ; IF CS, FAILURE
402             BIT      #TS$NHD,L$BFLG(RO) ; DOES IMAGE HAVE A HEADER?
403             BEQ      141$        ; IF EQ, YES - NOT ALLOWED
404             CMP      L$BLDZ(RO),L$BMXZ(RO) ; IS THE IMAGE OVERLAID?
405             BNE      141$        ; IF NE, YES - NOT ALLOWED
406             MOV      L$BSA(RO),BASE ; SAVE BASE ADDRESS FOR LATER
407             SUB      BASE,LENGTH ; ADJUST DEVICE TABLES SIZE
408             CLC                ; INDICATE SUCCESS
409             RETURN
410
411      ;
412      ; ERROR CONDITIONS
413
414      101$: EMSG$R 2F          ; OPEN FAILURE
415      121$: EMSG$R 2H          ; FILE NOT CONTIGUOUS
416      131$: MOVW    $IOSB,@%CLIOS ; COPY ERROR CODE
417      141$: EMSG$R 2J          ; LABEL BLOCK READ FAILURE
418      141$: EMSG$R 2K          ; INVALID BINARY IMAGE (HEADER OR OVERLAID)

```

VDEV CREATED BY MACRO ON 3-SEP-85 AT 10:28

PAGE 2

SYMBOL CROSS REFERENCE

CREF 04.00

SYMBOL	VALUE	REFERENCES
REP9	002346 R	7-157 7-161 7-162 7-163 7-164 7-165 7-166 7-167 7-168
RVAD	002004 R	#17-751 17-753 #13-625
R\$MPL	= *****	13-517 13-529
STBEXT	000000 R	8-236 13-547
SYSFDB	= ***** GX	#6-76 10-342
S\$BAS	= *****	*8-209 *8-210 *8-226 *8-233 *8-250 *8-254 *8-255 *9-281 *12-441
		*12-442 *12-445 *12-446 *13-575 *13-577 *15-688 *15-689 *15-691 *15-692
		7-148 7-148 7-149 7-149 7-150 7-150 7-151 7-151 7-152
		7-152 7-153 7-153 7-154 7-154 7-155 7-155 7-156 7-156
		7-157 7-157 7-158 7-158 7-159 7-159 7-160 7-160 7-161
		7-161 7-162 7-162 7-163 7-163 7-164 7-164 7-165 7-165
		7-166 7-166 7-167 7-167 7-168 7-168 7-169 7-169
S.LHD	= ***** GX	13-541
TABEND	000202 R	#6-140 *12-451 13-633
TS\$NHD	= ***** GX	11-402
UNITS	000172 R	#6-128 *8-252 *9-297 *13-510
U.CW3	000014	*8-234
U.DCB	000000	*13-534
U.RED	000002	*13-537
U.SCB	000020	13-524
VAD	002014 R	13-513 13-525 13-558 #13-627
VREL	000200 R	#6-135 *13-495 *13-496 13-499 13-521 13-533 13-536 13-539 13-544
		13-627 13-635
Z.PCB	= ***** GX	8-229
\$ALL15	= ***** GX	12-443
\$AZFS	= ***** GX	10-343 11-393
\$BFR	= ***** GX	8-254 8-255 12-441 12-442 15-688 15-689
\$CAT5	= ***** GX	10-331
\$CLDEQ	= ***** GX	7-154 7-155 7-156 7-157 8-223 15-693
\$CLIOS	= ***** GX	11-395 11-416 13-608
\$CLOPE	= ***** GX	11-394
\$CUR.N	= ***** GX	7-155 7-158
\$CSTA	= ***** GX	16-718 16-731 16-733
\$DEA16	= ***** GX	15-690
\$DEVHD	= ***** GX	9-278 13-583 *13-593
\$DFN2	= ***** GX	10-326
\$ERR2A	002204 R	#7-148 10-352
\$ERR2B	002226 R	6-110 #7-149
\$ERR2C	002264 R	6-111 #7-150
\$ERR2D	002322 R	6-112 #7-151
\$ERR2E	002350 R	6-113 #7-152
\$ERR2F	002376 R	#7-153 11-414
\$ERR2H	002434 R	#7-154 11-415
\$ERR2J	002464 R	#7-155 11-417
\$ERR2K	002536 R	#7-156 11-418
\$ERR2L	002600 R	#7-157 12-463
\$ERR2M	002634 R	#7-158 13-609
\$ERR2N	002672 R	#7-159 10-360
\$ERR2P	002706 R	#7-160 10-364
\$ERR2Q	002744 R	#7-161 9-304 13-610
\$ERR2R	003000 R	#7-162 13-611
\$ERR2S	003052 R	#7-163 13-614

VDIS - VNP SHOW COMMANDS
\$DSYS - SHO SYS

MACRO V05.03b Monday 15-Jul-85 12:14 Page 8

6.15

```
327 .SBTTL $DSYS - SHO SYS
328
329
330
331
332
333
334
335
336
337
338
339 001706 $DSYS:: CALL CKTYPE ; MOVE TYPE INTO STRING TO BE DISPLAYED
340 001712 103004 BCC 10$ ; BR IF SUCCESS
341 001714 ERRPT$ SYNERR,$EROUT ; ELSE SYNTAX ERROR
342 001724 10$: ERRPT$ MSG1,MSGOUT ; DISPLAY SYSTEM CHARACTERISTICS
343 001734 ERRPT$ MSG2,MSGOUT ;
344 001744 DIR$ #TIDE1 ; DETACH TERMINAL
345 001752 RETURN ; FINISHED
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
```

```

;
; DISPLAY MESSAGES
;
.ENABL LC
ERMSG$ <%NSystem,%A as of %Y%N>
ERMSG$ <%N Maximum control buffers = %D, Maximum small buffers = %D>
ERBLK$ MSG1,<TEMP1,$CCBNM,$SDBNM>

ERMSG$ < Maximum large buffers = %D, Large buffer size = %D>
ERMSG$ <%N Minimum receive buffers = %D%N>
ERBLK$ MSG2,<$RDBNM,$RDBSZ,$RDBTH>

;
; ERROR MESSAGE
;
ERMSG$ <Syntax error>
ERBLK$ SYNERR

.DSABL LC
```

H.15

```

867      .SBTTL DLONAM - DISPLAY LOGGING SINK NAME
868
869      ;+
870      DLONAM - DISPLAY LOGGING SINK NAME
871
872      INPUTS:
873      R4 - LOGGING STATE BIT
874      $OPTON - COMMAND OPTIONS
875
876      OUTPUTS:
877      LOGGING SINK NAME DISPLAYED ON USER'S TERMINAL
878      R0,R1,R2,R3 DESTROYED
879
880      -
881
882      004756 012700 001062' DLONAM: MOV #TEMP2,R0 ; POINT TO TEMPORARY STORAGE AREA
883      004762 032767 000400 000000G BIT #OP$CON,$OPTON ; SHOW CONSOLE NAME?
884      004770 001015 BNE 10$ ; BR IF YES
885      004772 032767 000200 000000G BIT #OP$FIL,$OPTON ; SHOW FILE NAME?
886      005000 001016 BNE 20$ ; BR IF YES
887
888      005002 016701 000000G MOV $LGMON,R1 ; ELSE SHOW MONITOR NAME
889      005006 CALL $C5TA ; CONVERT FIRST HALF OF MONITOR NAME
890      005012 016701 000002G MOV $LGMON+2,R1 ; POINT TO SECOND HALF OF NAME
891      005016 CALL $C5TA ; CONVERT IT
892      005022 000466 BR 60$ ; FINISH UP
893
894      005024 012703 000000G 10$: MOV #$LGCON,R3 ; POINT TO CONSOLE NAME
895      005030 CALL CVTDEV ; CONVERT IT TO ASCII
896      005034 000461 BR 60$ ; FINISH UP
897
898      005036 012703 000020G 20$: MOV #$LGFB+20,R3 ; POINT TO FILE DEVICE NAME
899      005042 CALL CVTDEV ; CONVERT IT TO ASCII
900      005046 112720 000133 MOVB #'[(R0)+ ; STORE LEFT BRACKET IN OUTPUT BUFFER
901      005052 116701 000001G MOVB $LGUC+1,R1 ; GET USER GROUP CODE
902      005056 005002 CLR R2 ; SUPPRESS ZERGES
903      005060 CALL $CBOMG ; CONVERT IT TO ASCII
904      005064 112720 000054 MOVB #'<,>,(R0)+ ; INSERT DELIMITER
905      005070 116701 000000G MOVB $LGUC,R1 ; GET USER MEMBER CODE
906      005074 005002 CLR R2 ; SUPPRESS ZEROES
907      005076 CALL $CBOMG ; CONVERT IT TO ASCII
908      005102 112720 000135 MOVB #'J,(R0)+ ; STORE RIGHT BRACKET IN OUTPUT BUFFER
909      005106 012703 000006G MOV #$LGFB+6,R3 ; POINT TO FILE NAME
910      005112 012746 000003 MOV #'-(SP) ; CONVERT THREE WORDS
911      005116 012301 30$: MOV (R3)+,R1 ; GET NEXT 3 CHARACTERS OF FILE NAME
912      005120 CALL $C5TA ; CONVERT THEM TO ASCII
913      005124 005316 DEC (SP) ; MORE TO CONVERT?
914      005126 003373 BGT 30$ ; BR IF YES
915      005130 005726 TST (SP)+ ; CLEAN UP STACK
916      005132 122740 000040 40$: CMPB #'-,(R0) ; DELETE TRAILING BLANKS
917      005136 001775 BEQ 40$ ;
918      005140 005200 INC R0 ; POINT PAST LAST NON-BLANK
919      005142 112720 000056 MOVB #'>,(R0)+ ; STORE PERIOD IN OUTPUT BUFFER
920      005146 012301 MOV (R3)+,R1 ; GET FILE TYPE
921      005150 CALL $C5TA ; CONVERT IT TO ASCII
922      005154 122740 000040 50$: CMPB #'-,(R0) ; DELETE TRAILING BLANKS
923      005160 001775 BEQ 50$ ; ...

```

```

245 000732 000000G .WORD $ERR53 ; FILE NOT CONTIGUOUS
246 000734 000000G .WORD $ERR54 ; LABEL BLOCK READ FAILURE
247 000736 000000G .WORD $ERR55 ; INVALID IMAGE (HEADER, OVERLAID, ETC.)
248 .IF DF M$$MGE
249 000740 000000G .WORD $ERR56 ; PROCESS NOT BUILD FOR APR5
250 000742 000000G .WORD $ERR57 ; PROCESS EXCEEDS 4K WORDS
251 000744 000000G .WORD $ERR58 ; PROCESS EXTENSION EXCEEDS 4K WORDS
252 .IFF
253 .WORD 101,101,101
254 .ENDC
255 000746 000000G .WORD $ERR59 ; PARTITION NOT IN SYSTEM
256 000750 000000G .WORD $ERR60 ; BASE ADDRESS MISMATCH
257 000752 000000G .WORD $ERR61 ; PARTITION TOO SMALL
258 000754 000000G .WORD $ERR62 ; NOT COMMON PARTITION
259 000756 000000G .WORD $ERR63 ; PARTITION BUSY
260 .IF DF M$$MGE
261 000760 000000G .WORD $ERR64 ; CEX PARTITION NOT IN EXEC SPACE
262 000762 000000G .WORD $ERR65 ; SUB-PCB ALLOCATION FAILURE
263 000764 000000G .WORD $ERR66 ; MAIN PARTITION TOO FRAGMENTED
264 .IFF
265 .WORD 101,101,101
266 .ENDC
267 000766 000000G .WORD $ERR67 ; PROCESS READ FAILURE
268 000770 000000G .WORD $ERR68 ; CETAB ALLOCATION FAILURE
269 000772 000000G .WORD $ERR9E ; TIMER SERVICE INIT FAILURE
270
271 000000 .PSECT

```

```
782 001622 RESRG <R0> ; RESTORE REG
783 001624 SEC ; INDICATE ERROR
784 001626 RETURN
785
786 001630 50$: GETRV R5,TLGTH ; REREAD TCB
787 001646 005767 000012G : ST T.RCVL+$BFR ; ANY PACKETS QUEUED ?
788 001652 001425 : BEQ 70$ ; BR IF NO
789 001654 : GETRV T.RCVL+$BFR,#DSIZE ; READ IN PACKET
790
791 : IF NDF R$$MPL ;[MP01]
792 001674 032767 000001 000006G : BIT #LS.CEX,LR.STS+6+$BFR ; IS SET CEX ALREADY QUEUED?
793 : ;[MP01]
794 : IF ;[MP01]
795 : BIT #LS.CEX,LR.STS+10+$BFR ; ...
796 : .ENDC ;[MP01]
797
797 001702 001401 : BEQ 60$ ; BR IF NO
798 001704 000467 : BR 110$ ; ELSE NOTHING TO DO
799 001706 052767 000001 000006G 60$: BIS #LS.CEX,LR.STS+6+$BFR ; ELSE SET CEX ALSO
800 001714 : PUTRC ; REWRITE PACKET
801 001720 : CALL $DEA16 ; DEALLOCATE PACKET
802 001724 000457 : BR 110$ ; AND CONTINUE
803 001726 010067 000012G 70$: MOV R0,T.RCVL+$BFR ; POINT LISHEAD AT FIRST PACKET
804 001732 010067 000014G : MOV R0,T.RCVL+2+$BFR ; ...
805 001736 : PUTRC ; REWRITE NTINIT TCB ADDRESS
806
807 001742 005067 000000G : CLR $BFR ; CLEAR LINK WORD ;[MP01]
808 : ;[MP01]
809 : IF NDF R$$MPL ;[MP01]
810
811 001746 012767 131574 000002G : MOV #*R...,$BFR+2 ; LOAD RAD50 NAME
812 001754 012767 105700 000004G : MOV #*RVNP,$BFR+4 ; ... VNP
813 001762 012767 000001 000006G : MOV #LS.CEX,LR.STS+6+$BFR ; SET CEX
814 001770 052767 020000 000006G : BIS #LS.UNF,LR.STS+6+$BFR ; SET FLAG THAT UNFIXES NTINIT
815 001776 016767 000000G 000040G : MOV ..COO,$BFR+32. ; ERRORS GO TO COO:
816 002004 012767 000401 000042G : MOV #401,$BFR+34. ; [1,1]
817
818 : IF ;[MP01]
819 : MOV #<13.+2>,$BFR+2 ; SET DATA LENGTH ;[MP01]
820 : MOV #*R...,$BFR+4 ; SET SENDER TASK NAME ;[MP01]
821 : MOV #*RVNP,$BFR+6 ; ...
822 : MOV #LS.CEX,$BFR+LR.STS+10 ; SET CEX ;[MP01]
823 : BIS #LS.UNF,$BFR+LR.STS+10 ; UNFIX NTINIT FLAG ;[MP01]
824 : MOV ..COO,$BFR+34 ; ERRORS TO CO: ;[MP01]
825 : MOV #401,$BFR+36. ; [1,1] ;[MP01]
826
827 : .ENDC ;[MP01]
828 : IF DF R$$MPL ;[MP01]
829 : BIT #DF.POL,$DFLAG ; SECONDARY POOL SUPPORTED ? ;[MP01]
830 : BEQ 80$ ; IF NO - BRANCH ;[MP01]
831 : MOV #DSIZE,R1 ; SET SIZE ;[MP01]
832 : PUTAD #120000,R1,R0 ; UPDATE PACKET ;[MP01]
833 : BR 90$ ; CONTINUE ;[MP01]
834 : 80$: PUTRC R0,R1 ; ... ;[MP01]
835
836 : IF ;[MP01]
837 002012 : PUTRC R0,R1 ; REWRITE NODE TO POOL ;[MP01]
838 : .ENDC
```

5-
6-
7-
8-
9-
10-
11-
12-
13-
14-
15-
16-
17-
18-
19-
20-
21-
22-
23-
24-
25-
32-
33-
34-

5-	55	MACRO DEFINITIONS	
6-	82	CONSTANT DEFINITIONS	
7-	105	LOCAL DATA	
8-	247	\$NLCFG - CONFIG FILE SCAN	
9-	445	OPEN CONFIG FILE	
10-	469	READ - READ NEXT RECORD	
11-	500	Q*** - LOOK FOR ***\$DF MACRO	
12-	554	CERR,CFERR - PRINT ERROR	
13-	593	STORID - Store id-string according to system type	;[TD001]
14-	655	MOVE - Move bytes	;[TD001]
15-	677	SPFILL - Fill area with spaces	;[TD001]
16-	699	CFLIN - ECHO LAST RECORD	
17-	713	SLT\$DF STATE TABLES	
18-	792	STA\$DF STATE TABLE	
19-	835	ADD\$DF STATE TABLE	
20-	853	DDM\$DF STATE TABLE	
21-	887	CNT\$DF STATE TABLE	
22-	928	UNT\$DF STATE TABLE	
23-	984	FEA\$DF STATE TABLE	
24-	1015	LLC\$DF STATE TABLE	
25-	1065	TPARS ACTION ROUTINES	
32-	1567	FEA\$DF ACTION ROUTINES	
33-	1598	LNKEND - LINK BLOCK AT END OF LIST	
34-	1624	FNDPDV - FIND PDV	

500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525
 526
 527
 528
 529
 530
 531
 532
 533
 534
 535
 536
 537
 538
 539
 540
 541
 542
 543
 544
 545
 546
 547
 548
 549
 550
 551
 552
 553
 554
 555
 556
 557
 558
 559
 560
 561
 562
 563
 564
 565
 566
 567
 568
 569
 570
 571
 572
 573
 574
 575
 576
 577
 578
 579
 580
 581
 582
 583
 584
 585
 586
 587
 588
 589
 590
 591
 592
 593
 594
 595
 596
 597
 598
 599
 600

```

500      .SBTTL Q*** - LOOK FOR ***$DF MACRO
501      +
502      Q*** - LOOK FOR *;
503
504      INPUTS:
505      NONE
506
507      OUTPUTS:
508      C-BIT=SUCCESS/FAILURE
509      R3,R4,R5=DESTROYED
510      -
511
512      .ENABL LSB
513
514      001064 012705 000356' QPLT: MOV #PLTDF,R5 ; STATE TABLE ADDRESS
515      001070 000436 BR 10$
516      001072 012705 000000' QSLT: MOV #SLTDF,R5 ; STATE TABLE ADDRESS
517      001076 005067 000000G CLR $SNBRA ; INIT NI ROUTER ADJACENCY VALUE
518      001102 005067 000000G CLR $SRPRI ; INIT ROUTER PRIORITY VALUE
519      001106 042767 000001 000136' BIC #FG.X2P,CFGFLG ; CLEAR X2P$DF SEEN
520      001114 000424 BR 10$
521      001116 012705 000244' QSTA: MOV #STADF,R5 ; STATE TABLE ADDRESS
522      001122 000421 BR 10$
523      001124 012705 000342' QADD: MOV #ADDDF,R5 ; STATE TABLE ADDRESS
524      001130 000416 BR 10$
525      001132 012705 000372' QDDM: MOV #DDMDF,R5 ; STATE TABLE ADDRESS
526      001136 000413 BR 10$
527      001140 012705 000456' QCNT: MOV #CNTDF,R5 ; STATE TABLE ADDRESS
528      001144 000410 BR 10$
529      001146 012705 000514' QUNT: MOV #UNTDF,R5 ; STATE TABLE ADDRESS
530      001152 000405 BR 10$
531      001154 012705 000662' OFEA: MOV #FEADF,R5 ; STATE TABLE ADDRESS
532      001160 000402 BR 10$
533      001162 012705 000724' QLLC: MOV #LLCDF,R5 ; STATE TABLE ADDRESS
534      001166 005001 10$: CLR R1 ; FULL KEYWORD MATCH LENGTH
535      001170 012702 000000' MOV #CFGKW,R2 ; KEYWORD TABLE ADDRESS
536      001174 016703 002700' MOV CFGSZ,R3 ; RECORD LENGTH
537      001200 012704 002472' MOV #CFGBF,R4 ; RECORD BUFFER ADDRESS
538      001204 005067 002704' CLR SYNERR ; CLEAR SYNTAX ERROR FLAG
539
540      001210 CALL TPARS ; GO DO THE PARSE
541      001214 103003 BCC 20$ ; IF CC, FOUND WHAT WE WERE LOOKING FOR
542      001216 005367 002704' DEC SYNERR ; DID SYNTAX ERROR OCCUR?
543      001222 001401 BEQ 101$ ; IF EQ, YES
544      001224 20$: RETURN
545
546
547      ;
548      ; ERROR CONDITION
549
550      001226 101$: MSG$R 1T ; SYNTAX ERROR
551
552      .DSABL LSB

```

928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982

VCFG -
TPARS

```

1503
1504      ; NO SECONDARY CSR (UNT$DF)
1505
1506 003536 032767 000000G 000000G U.XCSR: BIT #FL.LMC,$FLAGS ; COMIOP DEVICE?
1507 003544 001410          BEQ 10$ ; IF EQ, NO
1508 003546 032767 000000G 000000G BIT #FL.KMX,$FLAGS ; COMIOP MUX DEVICE?
1509 003554 001404          BEQ 10$ ; IF EQ, NO
1510 003556 032767 000007 000152' BIT #7,UCNT ; IS THIS THE FIRST, NINTH, ETC.?
1511 003564 001403          BEQ 101$ ; IF EQ, YES
1512 003566 005267 000152' 10$: JNC UCNT ; INCREMENT UNIT COUNT
1513 003572          RETURN
1514 003574          101$: MSG$R 1N ; SECONDARY CSR MISSING
1515
1516      ; UNIT COST (UNT$DF)
1517
1518
1519 003602 026727 000000G 000031 U.CST: CMP .PNUMB,#MAXCST ; IS THE COST IN RANGE ?
1520 003610 101006          BHI 101$ ; IF HI, NO .. ERROR
1521 003612 016700 000000G          MOV $SLTA,R0 ; ELSE, GET THE SLT ADDRESS
1522 003616 116760 000000G 000000G MOVB .PNUMB,L.COST(R0) ; STORE THE LINE COST
1523 003624          RETURN
1524 003626          101$: MSG$R YP ; ILLEGAL LINE COST
1525
1526      ; UNIT DEAD POLLING RATIO
1527
1528
1529 003634          U.DPR: MOV .PNUMB,$$DPR ; SAVE DEAD POLLING RATIO
1530 003634 016767 000000G 000000G RETURN
1531 003642
  
```

VCFG

SYMBOL

SYMBOL

\$ERRYS
 \$ERRZQ
 \$ERR1A
 \$ERR1B
 \$ERR1C
 \$ERR1D
 \$ERR1E
 \$ERR1F
 \$ERR1G
 \$ERR1H
 \$ERR1J
 \$ERR1K
 \$ERR1L
 \$ERR1M
 \$ERR1N
 \$ERR1P
 \$ERR1Q
 \$ERR1R
 \$ERR1S
 \$ERR1T
 \$ERR1U
 \$ERR1V
 \$ERR1W
 \$ERR1X
 \$ERR1Y
 \$ERR1Z
 \$ERR2A
 \$ERR2B
 \$ERR2C
 \$ERR2F
 \$ERR2I
 \$ERR3A
 \$ERR3B
 \$ERR3C
 \$EXIT
 \$FAIL
 \$FEATR
 \$FLAGS
 \$GCLEF
 \$GETAC
 \$GETRV
 \$GPRM
 \$LAMD
 \$MISS
 \$MXVCT
 \$NAME
 \$NLFCF
 \$NUMBF
 \$PDVNM
 \$PDVTA
 \$PFLAC

VCFG

VCFG CREATED BY MACRO ON 15-JUL-85 AT 19:06 PAGE 5 H 7
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
\$ERRYS	002106 R	#7-212
\$ERRZQ	001630 R	#7-206 25-1123
\$ERR1A	000216 R	#7-180 25-1226
\$ERR1B	000246 R	#7-181 25-1227
\$ERR1C	000306 R	#7-182 27-1291
\$ERR1D	000340 R	#7-183 27-1292
\$ERR1E	000376 R	#7-184 27-1304
\$ERR1F	000424 R	#7-185 27-1336
\$ERR1G	000460 R	#7-186 28-1363
\$ERR1H	000506 R	#7-187 28-1364
\$ERR1J	000544 R	#7-188
\$ERR1K	000600 R	#7-189 28-1387
\$ERR1L	000640 R	#7-190 28-1442
\$ERR1M	000700 R	#7-191 28-1443
\$ERR1N	000744 R	#7-192 30-1514
\$ERR1P	001004 R	#7-193 9-467
\$ERR1Q	001042 R	#7-194 10-497
\$ERR1R	001100 R	#7-195 10-498
\$ERR1S	001130 R	#7-196 8-435
\$ERR1T	001160 RG	#7-197 11-550
\$ERR1U	001206 R	#7-198 8-436
\$ERR1V	001246 R	#7-199 8-437
\$ERR1W	001306 R	#7-200 8-438
\$ERR1X	001346 R	#7-201 8-439
\$ERR1Y	001406 R	#7-202 8-363
\$ERR1Z	001456 R	#7-203 8-440
\$ERR2A	001516 R	#7-204 25-1192
\$ERR2B	001566 R	#7-205 25-1207
\$ERR2C	002312 RG	#7-216
\$ERR2F	002346 RG	#7-217
\$ERR2I	002402 RG	#7-218
\$ERR3A	002152 R	#7-213 8-441
\$ERR3B	002210 R	#7-214 8-442
\$ERR3C	002246 R	#7-215 8-443
\$EXIT	= 000000	#17-717
\$FAIL	= 177777	#17-717
\$FEATR	= ***** GX	*8-390 *32-1585 *32-1590 *32-1595 30-1506 30-1508
\$FLAGS	= ***** GX	28-1376 28-1379 28-1422 28-1425
\$GCLEP	= ***** GX	10-482
\$GETAD	= ***** GX	33-1617
\$GETRV	= ***** GX	33-1612
\$GPRM	= *****	17-717
\$LAMDA	= 000000	#17-717
\$MISS	= ***** GX	8-274 8-308 8-313 8-332 *8-340 *8-357 25-1096 25-1107 27-1280
\$MXVCT	= ***** GX	*27-1289 27-1297 *27-1302 27-1324 *27-1334 28-1435
\$NAME	= ***** GX	27-1286 8-267
\$NLCFG	= 000000 RG	#8-258
\$NUMBR	= 000002	#17-717
\$PDVNM	= ***** GX	34-1641
\$PDVTA	= ***** GX	34-1640 34-1652
\$PFLAG	= ***** GX	8-302 13-632

```

289 .SBTTL $NTECFP - CONFIG FILE SCAN
290
291 ;+ $NTECFP - PROCESS CONFIG FILE FOR PSI PARAMETERS
292
293 ; THIS ROUTINE IS CALLED DURING "SET CEX" TO PROCESS THE CONFIG FILE
294 ; FOR NTL PARAMETERS PERTAINING TO PSI.
295
296 INPUTS:
297 $SLTA=SLI ADDRESS
298
299 OUTPUTS:
300 C-BIT=SUCCESS/FAILURE
301
302 ;
303 $NTECFP::VNPDBG VCFP ; VNP debugging break point ;[TD001]
304 000000 CALL $CLSAV ; SAVE GCL CONTEXT ;[TD001]
305 000004 112767 000000G 000000G MOVB #CL.OPN,$CLFLG ; SET UP NECESSARY CONTEXT ;**2
306 000012 010667 000340 MOV SP,$PSAV ; SAVE SP FOR ERROR EXIT
307 000016 CALL OPEN ; OPEN THE CONFIG FILE
308 000022 10$: CALL READ ; READ NEXT RECORD
309 000026 103535 BCS 100$ ; IF CS, EOF (PSN$DF MISSING - NOTHING TO DO)
310 000030 CALL $QPSN ; IS IT PSN$DF ?
311 000034 103772 BCS 10$ ; BR IF NO .. KEEP LOOKING
312 000036 032767 000000G 000000G 20$: BIT #PF.XGA,$PFLAG ; If host system, X3P$DF may be absent ... ;[TD001]
313 000044 001006 BNE 30$ ; ... in which case do not check here ;[TD001]
314 000046 CALL READ ; READ NEXT RECORD ;[TD001]
315 000052 103533 BCS 101$ ; IF CS, EOF (X3P$DF MISSING) ;**1
316 000054 CALL $QX3P ; IS IT X3P$DF?
317 000060 103766 BCS 20$ ; BR IF NO .. KEEP LOOKING
318
319 ; READ NEXT RECORD FROM CONFIG FILE
320
321 000062 30$: CALL READ ; READ NEXT RECORD
322 000066 103515 BCS 100$ ; BR IF EOF ;[TD001]
323
324 ; If host system, check for an X3P$DF ;[TD001]
325 ;[TD001]
326 000070 032767 000000G 000000G 32$: BIT #PF.XGA,$PFLAG ;[TD001]
327 000076 001403 BEQ 34$ ; No check if not host ;[TD001]
328 000100 CALL $QX3P ; Check for X3P$DF ;[TD001]
329 000104 103366 BCC 30$ ; Look for next macro if found ;[TD001]
330
331 ; LOOK FOR DTE$DF, DST$DF, RDT$DF OR X29$DF
332
333 000106 34$: CALL $QDTE ; IS IT DTE$DF? ;[TD001]
334 000112 103031 BCC 35$ ; BR IF YES ;**1
335 000114 CALL $QDST ; IS IT DST$DF?
336 000120 103047 BCC 70$ ; BR IF YES
337 000122 CALL $QRDT ; IS IT RDT$DF?
338 000126 103355 BCC 30$ ; BR IF YES - LOOK FOR NEXT
339 000130 CALL $QX29 ; IS IT X29$DF?
340 000134 103352 BCC 30$ ; Branch if yes ;[TD001]
341 ;[TD001]
342 ; If host system, check for remote network names ;[TD001]
343 ;[TD001]
344 000136 032767 000000G 000000G BIT #PF.XGA,$PFLAG ;[TD001]
345 000144 001746 BEQ 30$ ; If not host, ignore RNW$DF ;[TD001]

```

```

698 .SBTTL STORID - Store id-string according to system type ;[TD001]
699 ;+ ;[TD001]
700 *** STORID - Store id-string according to system type ;[TD001]
701 ;[TD001]
702 FUNCTION ;[TD001]
703 ;[TD001]
704 This routine is used to check and store various parameters ;[TD001]
705 during the parsing of CETAB macros. In PSI V2.0 compatible ;[TD001]
706 systems, it checks for and stores parameters in PSI V2.0 ;[TD001]
707 compatible format. In extended systems, id-strings of up ;[TD001]
708 to MAXID characters are permitted and stored appropriately. ;[TD001]
709 The buffer is padded with spaces. The length returned in the ;[TD001]
710 buffer is the maximum length for old-format systems (fixed ;[TD001]
711 fields) or the actual length for new-format systems (variable ;[TD001]
712 fields). ;[TD001]
713 ;[TD001]
714 INPUTS ;[TD001]
715 ;[TD001]
716 R0 - address of buffer to contain string prefixed by length ;[TD001]
717 R1 - maximum length of string in old-format system ;[TD001]
718 (R1 < MAXID) ;[TD001]
719 .PSTCN - actual length of string ;[TD001]
720 $PFLAG - PSI system type flags ;[TD001]
721 ;[TD001]
722 OUTPUTS ;[TD001]
723 ;[TD001]
724 R0, R1, R2 destroyed ;[TD001]
725 String and length stored in buffer ;[TD001]
726 Carry set if invalid string length for system type ;[TD001]
727 ;[TD001]
728 ;[TD001]
729 - ;[TD001]
730 STORID:: ;[TD001]
731 MOV R0, -(SP) ; Save buffer address ;[TD001]
732 MOV R1, -(SP) ; Save maximum length ;[TD001]
733 CLRB (R0)+ ; Clear length field in output ;[TD001]
734 MOV #MAXID, R1 ; Always clear MAXID spaces ;[TD001]
735 CALL SPFILL ;[TD001]
736 MOV (SP)+, R1 ; Restore length ;[TD001]
737 MOV (SP)+, R2 ; Get back buffer address ;[TD001]
738 BIT #PF.XDF, $PFLAG ; Check for extended data format ;[TD001]
739 BNE 20$ ;[TD001]
740 ;[TD001]
741 CMP .PSTCN, R1 ; Old format, so use max length in R1 ;[TD001]
742 BHI 99$ ; HI, then string too long ;[TD001]
743 MOVB R1, (R2)+ ; Set length to maximum for fixed field ;[TD001]
744 BR 30$ ;[TD001]
745 ;[TD001]
746 CMP .PSTCN, #MAXID ; New format, max length is MAXID ;[TD001]
747 BHI 99$ ;[TD001]
748 MOVB .PSTCN, (R2)+ ; Set actual length for variable field ;[TD001]
749 ;[TD001]
750 ; Move string to buffer ;[TD001]
751 ;[TD001]
752 30$: MOV .PSTCN, R1 ; Get address of source ;[TD001]
753 MOV .PSTPT, R0 ; Move string to buffer ;[TD001]
754 CALL MOVE ; Signal success ;[TD001]
755 CLC ;[TD001]

```

.TITLE VC10 - LOAD CETAB IMAGE
.IDENT /V05.00/

COPYRIGHT (C) 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - LOAD CETAB IMAGE ROUTINES

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0

```
5.00 22-JUL-85
      DECnet-11M/S V4.2
      DECnet-11M-Plus V3.0
      DECnet-Micro/RX V1.0
```

★ ★ F I L

VV
VV
VV
VV
VV
VV
VV
VV
VV
VV
VV

[illegible]

[illegible]

```

LL          SSSSSSSS  TTTT TTTT TTTT
LL          SSSSSSSS  TTTT TTTT TTTT
          SS          TT
          SS          TT
          SS          TT
          SS          TT
          SSSSSS     TT
          SSSSSS     TT
          SS         TT
          SS         TT
          SS         TT
          SS         TT
          SSSSSSSS   TT
          SSSSSSSS   TT

```



```

420                                     .SBTTL  DVALL - ALLOCATE DEVICE TABLE
421
422                                     ;+
423                                     DVALL - ALLOCATE DEVICE TABLE SPACE FROM THE POOL
424                                     THIS ROUTINE IS CALLED TO ALLOCATE SPACE FOR THE DEVICE TABLES FROM THE
425                                     CORE POOL. THE AMOUNT TO ALLOCATE IS DEFINED BY THE VALUE OF $XXEND FROM
426                                     THE STB FILE. THE SYMBOL VALUES FROM THE STB FILE ARE ADJUSTED BY ADDING
427                                     THE ADDRESS OF THE ALLOCATED DATA.
428
429                                     INPUTS:
430                                     LENGTH=BYTE COUNT NEEDED
431
432                                     OUTPUTS:
433                                     C-BIT=SUCCESS/FAILURE
434                                     $LBUF=BUFFER ADDRESS (IF SUCCESSFUL)
435                                     $LLEN=DEVICE TABLES LENGTH
436
437                                     -
438                                     DVALL:
439                                     MOV     LENGTH,R1          ; GET BYTE COUNT NEEDED
440                                     ADD     #3,R1              ; ROUND UP TO NEXT MULTIPLE OF 4
441                                     BIC     #3,R1              ; IN MEMORY OF THE RPO* SERIES
442                                     SWTCHO          ; DEFAULT TO $BFR FOR $ALL15 ROUTINE
443                                     SWTCHI          ; DEFAULT TO $BFR FOR $ALL15 ROUTINE
444                                     CALL     $ALL15          ; TRY TO ALLOCATE FROM THE POOL
445                                     BCS     101$             ; IF CS, FAILURE
446                                     SWTCHO #DEVBUF          ; RETURN TO LOCAL BUFFER
447                                     SWTCHI #DEVBUF          ; RETURN TO LOCAL BUFFER
448                                     MOV     R0,PBUF          ; STORE ADDRESS AND
449                                     ADD     R0,R1              ; LENGTH
450                                     DEC     R1                ; CALCULATE END OF TABLES-1
451                                     MOV     R1,TABEND          ; SAVE IT FOR LATER
452                                     SUB     BASE,R0           ; ADJUST FOR 1KB BASE ADDRESS
453                                     MOV     #DBLK2,R1          ; START AT SECOND CONTROL BLOCK
454                                     ADD     R0,10(R1)          ; ADJUST DEVICE TABLE SYMBOL
455                                     MOV     (R1),R1           ; GET NEXT CONTROL BLOCK ADDRESS
456                                     BNE     30$                ; IF NE, NOT AT END
457                                     CLC                          ; INDICATE SUCCESS
458                                     RETURN
459
460                                     ;
461                                     ; ERROR CONDITION
462
463                                     101$:  MSG$R 2L              ; ALLOCATION FAILURE
  
```

VDEV CREATED BY MACRO ON 3-SEP-85 AT 10:28 PAGE 3 H 14
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
\$ERR2I	003104 R	#7-164 13-615
\$ERR2U	003156 R	#7-165 13-616
\$ERR2V	003230 R	#7-166 13-619
\$ERR2W	003262 R	#7-167 13-620
\$ERR2X	003334 R	#7-168 9-303
\$ERR2Y	003376 R	#7-169 8-262
\$FEATR	= ***** GX	8-234
\$GETRV	= ***** GX	8-226 8-233 9-281 13-587 13-596 13-600
\$IOSB	= ***** GX	11-416 13-608
\$LBUF	= ***** GX	13-626 15-686
\$LLEN	= ***** GX	*13-489 15-687
\$NAME	= ***** GX	10-324 16-717
\$NLDEV	000000 RG	#8-206
\$PDVA	= ***** GX	8-224
\$PRV.N	= ***** GX	7-149 7-150 7-153
\$PUTRC	= ***** GX	8-232 8-250 13-569 13-598 13-602
\$RADDR	= ***** GX	*8-226 *8-233 *8-250 *9-279 9-284 *13-569 *13-582 13-584 *13-585
		13-590 *13-596 *13-600
\$READ	= ***** GX	13-491
\$REP.C	= ***** GX	7-154
\$REP.P	= ***** GX	7-151 7-152 16-709
\$RLBL	= ***** GX	11-400
\$RSIZE	= ***** GX	*13-565 *13-568 *13-580
\$SAVAL	= ***** GX	14-650
\$SYMBL	= ***** GX	10-345
\$TIOUT	= ***** GX	14-667
\$SDEV	= ***** GX	9-282 10-327 10-328 13-500 17-754 17-755
\$TBL	= ***** GX	8-227

VDIS - VNP SHOW COMMANDS
\$DOBJ - SHO OBJ

MACRO V05.03b Monday 15-Jul-85 12:14 Page 9

H.15

367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423

002272
002276 103004
002300
012701 001062'
012705 000012
012703 000000G
005767 000000G
002330 001416
005703
002334 001446
012700 000037'
002342 112021
002344 001376

002346 005004
002350
002354 103436
002356
002362 005204
002364 000771

002366 005703
002370 001421
002372 012700 000055'
002376 112021
002400 001376
002402
002406 103412
002410 012702 000000G
002414 126263 000002 000002
002422 001367
002424 005004
002426
002432 000407

.SBTTL \$DOBJ - SHO OBJ

;\$DOBJ - SHOW KNOWN OBJECTS CHARACTERISTICS OR SUMMARY
SHOW OBJECT X CHARACTERISTICS OR SUMMARY

INPUTS: NONE

OUTPUTS: THE OBJECT CHARACTERISTICS OR SUMMARY ARE DISPLAYED
ON THE USERS TERMINAL

ALL REGISTERS ARE DESTROYED

```

$DOBJ:: CALL CKTYPE          ; GET COMMAND TYPE
          BCC 5$              ; BR IF SUCCESS
          ERRPT$ SYNERR,$EROUT ; ELSE SYNTAX ERROR
5$: MOV #TEMP2,R1             ; STORAGE FOR INTRO DISPLAY
          MOV #O.LEN,R5        ; GET LENGTH OF OBJECT BLOCK TO READ IN
          MOV #SOBJHD,R3       ; GET OBJECT LISTHEAD
          TST $QUAL            ; SHOW KNOWN OBJECTS?
          BEQ 30$              ; BR IF NO
          TST R3               ; ANY OBJECTS IN SYSTEM
          BEQ 70$              ; BR IF NO
          MOV #OKNOWN,R0       ; DISPLAY KNOWN OBJECTS
10$: MOVB (R0)+,(R1)+         ; STORE IT
          BNE 10$              ; ...

          ; SHOW KNOWN OBJECTS
20$: CLR R4                   ; INDICATE HEADER TO BE PRINTED
          CALL NXTBLK          ; GET NEXT ENTRY IN OBJECT TABLE
          BCS 70$              ; BR IF AT END OF OBJECT LIST
          CALL DISOBJ          ; DISPLAY THE OBJECT INFORMATION
          INC R4               ; INDICATE HEADER NOT DISPLAYED
          BR 20$               ; GO DO NEXT ONE IN LIST

          ; SHOW SPECIFIED OBJECT
30$: TST R3                   ; ANY OBJECTS IN SYSTEM
          BEQ 60$              ; BR IF NO
          MOV #OSPEC,R0        ; DISPLAY SPECIFIED OBJECT
40$: MOVB (R0)+,(R1)+         ; STORE IT
          BNE 40$              ; ...
50$: CALL NXTBLK              ; READ IN NEXT ENTRY IN OBJECT TABLE
          BCS 60$              ; BR IF AT END OF LIST
          MOV #R0B,R2          ; GET INPUT PARAMETERS
          CMPB LR.TYP(R2),O.TYP(R3) ; IS THIS THE ONE TO DISPLAY?
          BNE 50$              ; BR IF NO
          CLR R4               ; INDICATE PRINT HEADER
          CALL DISOBJ          ; DISPLAY OBJECT INFORMATION
          BR 70$               ; FINISHED

          ; SPECIFIED OBJECT NOT FOUND

```

VDIS - VNP SHOW COMMANDS
 DLONAM - DISPLAY LOGGING SINK NAME

MACRO V05.03b Monday 15-Jul-85 12:14 Page 15-1

H 16

924 005162 005200
 925 005164 112720 000073
 926 005170 012301
 927 005172 005002
 928 005174
 929 005200 105010
 930 005202
 931 005212

60\$:
 INC R0
 MOVB #*,(R0)+
 MOV (R3)+,R1
 CLR R2
 CALL \$CBOMG
 CLRB (R0)
 ERRT\$ LOMSG7,MSGOUT
 RETURN

; POINT PAST LAST NON-BLANK
 ; STORE SEMI-COLON
 ; GET UNIT NUMBER
 ; SUPPRESS LEADING ZEROES
 ; CONVERT IT TO ASCII
 ; CREATE ASCII STRING
 ; DISPLAY LOGGING SINK NAME MESSAGE

```
273 .SBTTL $$CEX - TRANSFER COMM EXEC
274
275 + $SCEX - TRANSFER COMM EXEC INTO CORE
276
277 INPUTS:
278 $$MUX=NETLDR DATA BLOCK ADDRESS
279
280 OUTPUTS:
281 C-BIT=SUCCESS/FAILURE
282 .CXSYM=PROCESS VALIDATION SYMBOL DEFINITIONS
283 .CXPCB=COMM EXEC PCB ADDRESS
284 .CXALL=CETAB ALLOCATION ADDRESS
285 .CXLEN=CETAB ALLOCATION LENGTH
286 $NTLPT=NETLDR DATA BLOCK ADDRESS
287 FE.CEX OF $FMASK SET IF SUCCESSFUL
288
289 $SCEX::
290 .MCALL VNPBPT
291 VNPBPT VCEX
292 CALL CESYM ; LOOK FOR SPECIAL SYMBOLS
293 BCC 1$ ; IF CS, ERROR
294 RETURN
295
296 000000 012700 000046 1$: MOV #46,R0 ; ASSUME ERROR (46=NTINIT INIT FAILURE)
297 000014 CALL RUNNTI ; KICK OFF TIMER SERVICE
298 000020 103002 BCC 10$ ; IF SUCCESS - BRANCH
299 000022 CALLR TSKERR ; EXIT ON ERROR
300 000026 012700 000000G 10$: MOV $DFUIC,R0 ; SETUP ASCIZ FILE SPEC
301
302 .IF NDF R$$MPL
303 000032 012701 000376' MOV #CEXNAM,R1 ; FOR [XXX,5X]CEX.STB
304 .IFF
305 MOV #RSX,R1 ; FOR [XXX,54]RSX11M.STB
306 .ENDC
307
308 000036 005002 CLR R2 ; ...
309 000040 CALL $AZFS ; ...
310
311 .IF NDF R$$MPL
312
313 CALL $PRID ; TRANSFER COMM EXEC INTO CORE
314 000050 103002 BCC 15$ ; IF SUCCESS - BRANCH
315 000052 CALLR TSKERR ; IF NOT - ERROR
316 000056 010167 000406' 15$: MOV R1,CEXPCB ; SAVE THE COMM EXEC PCB ADDRESS
317 000062 016700 000000G MOV #CEAVL,R0 ; GET ADDRESS OF "$CEAVL"
318 000066 GETRV R1,PLGTH ; READ IN THE COMM EXEC PCB
319 000104 012701 000000G MOV #BFR,R1 ; POINT AT THE INTERNAL PCB
320 000110 016102 000016 MOV P.BLK$(R1),R2 ; GET THE SIZE OF THE COMM EXEC PARTITION
321 000114 066102 000014 ADD P.REL(R1),R2 ; CALCULATE END OF PARTITION
322 000120 ASL$ 6,R2 ; CONVERT BLOCKS TO BYTES
323 000134 162700 000002 SUB #2,R0 ; POINT AT START OF POOL LISTHEAD
324 000140 010001 MOV R0,R1 ; SAVE START ADDRESS OF POOL LISTHEAD
325 000142 GETRV R0,#20 ; READ IN THE POOL LISTHEAD
326 000160 012700 000000G MOV #BFR,R0 ; POINT AT LISTHEAD TEMPLATE
327 000164 062701 000006 ADD #6,R1 ; POINT AT END OF LISTHEAD ( START OF FREE SPACE)
328 000170 061001 ADD (R0),R1 ; ROUND ADDRESS
329 000172 042001 BIC (R0)+,R1 ; ...
```

```

839 002026          90$:      MOV      #TICKS,R4      ; NUMBER OF TICKS
840 002026          012704 000001      CALL    RUN      ; RUN NTINIT
841 002032          BCC     100$      ; IF SUCCESS - BRANCH
842 002036          103007      ERRPT$ ERR4,MSGOUT    ; NO NODES IN POOL
843 002040          RESRG  <R0>      ; RESTORE REG
844 002050          SEC      ; INDICATE ERROR
845 002052          000261      RETURN
846 002054
847
848 002056          100$:  SWTCHO      ; POINT BACK AT DEFAULT BUFFER
849
850 002064          110$:  RESRG  <R0>      ; RESTORE REG
851 002066          RETURN
852

```

;[EMP01]

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52

.TITLE VCFG - VNP SCAN CONFIGURATION FILE
.IDENT /V05.00/

.. COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
.. DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

.. THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
.. ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
.. INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
.. COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
.. OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
.. TRANSFERRED.

.. THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
.. AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
.. CORPORATION.

.. DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
.. SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

.. MODULE DESCRIPTION:

.. VNP - CONFIG FILE SCANNER FOR "SET LINE" AND "SET PROCESS"

.. DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

.. IDENT HISTORY:

- .. 1.00 27-FEB-78
.. VERSION 2.0 RELEASE
- .. 2.00 14-DEC-79
.. DECNET-11M/S V3.0
.. DECNET-11M-PLUS V1.0
- .. 3.00 16-APR-82
.. DECNET-11M V3.1
.. DECNET-11M-PLUS V1.1
- .. 4.00 07-NOV-83
.. DECNET-11M V4.0
.. DECNET-11M-PLUS V2.0
- .. 5.00 22-JUL-85
.. DECnet-11M/S V4.2
.. DECnet-11M-Plus V3.0
.. DECnet-Micro/RSX V1.0

```

554      .SBTTL  CERR,CFERR - PRINT ERROR
555
556      ;+
557      CERR - PRINT ERROR MESSAGE
558      CFERR - PRINT FILE ASSOCIATED ERROR MESSAGE
559
560      INPUTS:
561      RO=MESSAGE BLOCK ADDRESS
562
563      OUTPUTS:
564      C-BIT=SET
565      RETURN TO $NLCFG'S CALLER
566
567      CERR::  MOV     RO,-(SP)      ; SAVE RO
568              CALL    $CLDEQ      ; CLOSE THE CONFIG FILE
569              MOV     (SP)+,RO     ; RESTORE RO
570      CFERR::  MOV     RO,R5       ; COPY ERROR MESSAGE BLOCK ADDRESS
571              MOV     #ERRBUF,R4   ; PLACE TO BUILD ERROR MESSAGE
572              MOV     (R5)+,R3     ; GET FORMAT STRING ADDRESS
573              CALL    @R5+         ; DO THE FORMATTING
574              TSTB    (R3)         ; IS IT END OF FORMAT STRING?
575              BEQ     20$          ; IF EQ, YES
576      10$:     MOVB    (R3)+,(R4)+  ; ELSE COPY REST OF FORMAT STRING
577              BNE     10$
578              DEC     R4           ; SKIP THE ZERO BYTE
579      20$:     MOV     (R5)+,R3     ; GET LINE 2 ADDRESS
580      30$:     MOVB    (R5)+,(R4)+  ; COPY THE ERROR TEXT
581              BNE     30$
582              DEC     R4           ; SKIP THE ZERO BYTE
583              MOV     #ERRMSG,RO   ; GET ERROR MESSAGE ADDRESS
584              MOV     R4,R1        ; COPY END ADDRESS
585              SUB     R0,R1        ; GIVING LENGTH
586              CALL    $TIOUT       ; PRINT IT
587              TST     R3           ; SHOULD SECOND LINE BE PRINTED?
588              BEQ     40$          ; IF EQ, NO
589              CALL    (R3)         ; PRINT SECOND ERROR MESSAGE LINE
590      40$:     SEC          ; RETURN FROM $NLCFG WITH C-BIT SET
591              MOV     SP,SP,SP     ; ...
              RTSPC::  RETURN
  
```

984
 985
 986
 987
 988
 989
 990
 991
 992
 993
 994
 995
 996
 997
 998
 999
 1000
 1001
 1002
 1003
 1004
 1005
 1006
 1007
 1008
 1009
 1010
 1011
 1012
 1013


```

: .SBTTL FEADF STATE TABLE
:
: FEATURES DEFINITION (FEADF)
:
: STATES$ FEADF
:   TRANS$ %FEADF%,.,1,SYNRR
:
: STATES$
:   TRANS$ $ALPHA,,F.CH1
:
: STATES$
:   TRANS$ $ALPHA,,F.CH2
:
: STATES$
:   TRANS$ <','>
:
: STATES$
:   TRANS$ $NUMBR,,F.FEA
:
: STATES$
:   TRANS$ <','>
:
: STATES$
:   TRANS$ $NUMBR,,F.MOD
:
: STATES$
:   TRANS$ <','>
:
: STATES$
:   TRANS$ $NUMBR,$EXIT,F.SET

```

```

1533
1534
1535
1536
1537 003644 005767 000000G U.HTIM: TST .PNUMH ; DOUBLE WORD VALUE?
1538 003650 001004 BNE 101$ ; BR IF YES - ERROR
1539 003652 016767 000000G 000000G MOV .PNUMB,$$HTIM ; SAVE HELLO TIMER VALUE
1540 003660 RETURN
1541
1542 003662 101$: MSG$R YA ; ILLEGAL HELLO TIMER VALUE
1543
1544
1545
1546
1547 003670 005767 000000G U.LTIM: TST .PNUMH ; DOUBLE WORD VALUE?
1548 003674 001004 BNE 101$ ; BR IF YES - ERROR
1549 003676 016767 000000G 000000G MOV .PNUMB,$$LTIM ; SAVE LISTEN TIMER VALUE
1550 003704 RETURN
1551
1552 003706 101$: MSG$R YB ; ILLEGAL LISTEN TIMER VALUE
1553
1554
1555
1556
1557
1558 003714 116767 000000G 000000G L.CH1: MOVB .PCHAR,$$DEV ; COPY THE CHARACTER
1559 003722 RETURN
1560
1561
1562
1563
1564 003724 116767 000000G 000001G L.CH2: MOVB .PCHAR,$$DEV+1 ; COPY SECOND CHARACTER
1565 003732 RETURN

```

VCFG

SYMBOL

SYMBOL

\$QDTE

\$QPV

\$QSV

\$QX2P

\$QX3P

\$RADDR

\$RAD50

\$RDBSZ

\$REP.N

\$RONLY

\$SLTA

\$STRNG

\$SUBXP

\$TIOUT

\$XLINK

\$APR

\$CSR

\$DCHA

\$DEV

\$DPR

\$EXT

\$HTIM

\$INC

\$LSAD

\$LTIM

\$MTP

\$MUX

\$NBRA

\$PCHA

\$PRI

\$RPR

\$SCSF

\$SNUM

\$VECT

\$FLC

\$KEY

.MGMGE

.PCHAR

.PNUMB

.PNUMH

.PSTC

.PSTP

.TPAR

..SCSF

VCFG

SYMBOL	VALUE	REFERENCES							
\$QDTE	= ***** GX	8-300							
\$QPVC	= ***** GX	8-304							
\$QSV	= ***** GX	8-296							
\$QX2P	= ***** GX	8-369							
\$QX3P	= ***** GX	8-298	8-397	8-414					
\$RADDR	= ***** GX	*33-1612	*33-1617						
\$RAD50	= 000016	#17-717							
\$RDBSZ	= ***** GX	29-1478							
\$REP.N	= ***** GX	7-193	7-194						
\$RONLY	= ***** GX	17-717	17-717						
\$SLTA	= ***** GX	8-269	17-1234	28-1372	28-1396	28-1411	28-1434	29-1469	30-1521
\$STRNG	= 000004	#17-717							
\$SUBXP	= 000010	#17-717							
\$TIOUT	= ***** GX	12-585	16-711						
\$XLINK	= ***** GX	33-1621							
\$APR	= ***** GX	*25-1255							
\$CSR	= ***** GX	*27-1301							
\$DCHA	= ***** GX	*28-1382	*28-1385	*28-1399	*28-1402	29-1458	29-1462	29-1473	
\$DEV	= ***** GX	*31-1558	*31-1564	32-1571	32-1578				
\$DPR	= ***** GX	*30-1530							
\$EXT	= ***** GX	*25-1109	*25-1118						
\$HTIM	= ***** GX	*25-1176	*26-1269	*31-1539					
\$INC	= ***** GX	*25-1098							
\$LSAD	= ***** GX	*25-1190							
\$LTIM	= ***** GX	*25-1178	*26-1271	*31-1549					
\$MTP	= ***** GX	*25-1203	*25-1223						
\$MUX	= ***** GX	8-338							
\$NBRA	= ***** GX	*11-317	*25-1146						
\$PCHA	= ***** GX	*28-1414	*28-1416						
\$PRI	= ***** GX	*27-1333							
\$RPR1	= ***** GX	*11-518	*25-1155						
\$SCSR	= ***** GX	*28-1439							
\$SNUM	= ***** GX	*25-1220							
\$VECT	= ***** GX	*27-1288							
\$FLG	= 177777	#17-717							
\$KEY	= 177777	#17-717							
.MGMGE	= ***** GX	25-1094	25-1105	25-1116					
.PCHAR	= ***** GX	31-1558	31-1564	32-1571	32-1578				
.PNUMB	= ***** GX	25-1098	25-1109	25-1118	25-1129	25-1137	25-1146	25-1153	25-1155
		25-1167	25-1176	*25-1177	25-1178	25-1188	25-1190	25-1199	25-1201
		25-1215	25-1220	25-1232	25-1243	25-1251	25-1255	26-1269	*26-1270
		27-1284	27-1286	27-1288	27-1313	27-1316	27-1328	27-1330	28-1358
		28-1373	28-1399	28-1402	28-1414	28-1416	29-1452	*29-1498	*29-1499
		30-1522	30-1530	31-1539	31-1549	32-1585	32-1590	32-1595	35-1661
		25-1165	25-1174	25-1186	25-1213	26-1264	27-1282	27-1309	27-1311
.PNUMH	= ***** GX	28-1356	28-1369	28-1392	28-1408	31-1537	31-1547	35-1659	35-1673
		13-635	13-640	13-642	13-646	25-1075			
.PSTCN	= ***** GX	13-647	25-1074						
.PSTPT	= ***** GX	11-540							
.TPARS	= ***** GX	28-1433							
..SCSR	= ***** GX								

```

346 000146          CALL    $ORNW          ; Check for RNW$DF          ;[TD001]
347 000152 103743   BCS     30$           ; Not RNW$DF, so keep Looking ;[TD001]
348 000154          CALL    READ           ; Check for matching RNA$DF  ;[TD001]
349 000160 103504   BCS     105$          ; RNA$DF missing if EOF     ;[TD001]
350 000162          CALL    $ORNA          ; Check for RNA$DF          ;[TD001]
351 000166 103501   BCS     105$          ; Not RNA$DF, so error      ;[TD001]
352 000170          CALL    RNWEND         ; End of remote network macros ;[TD001]
353 000174 000732   BR      30$           ; Look for another macro    ;[TD001]
354          ;                               ;[TD001]
355          ; Process DTE$DF macro and associated macros ;[TD001]
356          ; As PVC circuit blocks are allocated from separate PSI pool ;[TD001]
357          ; in gateway systems, PVC$DF is ignored for gateways. ;[TD001]
358          ;                               ;[TD001]
359 000176          35$: CALL    READ           ; Read next record          ;[TD001]
360 000202 103447   BCS     100$          ; Branch if eof            ;[TD001]
361 000204          CALL    $OCHN          ; Is it CHN$DF?            ;[TD001]
362 000210 103372   BCC     35$           ; Branch if yes - check for more ;[TD001]
363 000212 032767 000000G 000000G   BIT    #PF.PSV,$PFLAG ; If Gateway system, ignore PVC$DF macros ;[TD001]
364 000220 001003   BNE     36$           ; ... as nowhere to allocate circuit blocks ;[TD001]
365 000222          CALL    $OPVC          ; Is it PVC$DF?            ;[TD001]
366 000226 103363   BCC     35$           ; Branch if yes - check for more ;[TD001]
367 000230          36$: CALL    $OCUG          ; Is it CUG$DF?            ;[TD001]
368 000234 103360   BCC     35$           ; Branch if yes - check for more ;[TD001]
369 000236 000714   BR      32$           ; Else back to top level scan ;[TD001]
370          ;                               ;[TD001]
371          ; Process DST$DF, DSA$DF, DSC$DF, DSN$DF macros ;[TD001]
372          ;                               ;[TD001]
373 000240          70$: CALL    READ           ; READ NEXT RECORD          ;[TD001]
374 000244 103441   BCS     102$          ; BR IF EOF - NO DSA$DF    ;[TD001]
375 000246          CALL    $QDSA          ; IS IT DSA$DF?            ;[TD001]
376 000252 103436   BCS     102$          ; BR IF NO - NO DSA$DF     ;[TD001]
377 000254          CALL    READ           ; READ NEXT RECORD          ;[TD001]
378 000260 103436   BCS     103$          ; BR IF EOF - NO DSC$DF    ;[TD001]
379 000262          CALL    $QDSC          ; IS IT DSC$DF?            ;[TD001]
380 000266 103433   BCS     103$          ; BR IF NO - ERROR         ;[TD001]
381 000270 032767 000000G 000000G   BIT    #PF.PSV,$PFLAG ; If gateway system, insist on DSN$DF ;[TD001]
382 000276 001406   BEQ     72$           ; EQ, then not gateway system ;[TD001]
383 000300          CALL    READ           ;                               ;[TD001]
384 000304 103427   BCS     104$          ; EOF - No DSN$DF          ;[TD001]
385 000306          CALL    $QDSN          ; Check for DSN$DF         ;[TD001]
386 000312 103424   BCS     104$          ; Error if not present      ;[TD001]
387 000314          72$: CALL    DSTEND         ; END OF DESTINATION DESCRIPTOR BLOCK MACROS ;[TD001]
388 000320 000660   BR      30$           ; GO READ AND PROCESS NEXT  ;[TD001]
389          ;                               ;[TD001]
390          ; ALL NEEDED PARAMETERS SEEN ;[TD001]
391          ;                               ;[TD001]
392          ;                               ;[TD001]
393 000322 126727 000000G 000000G 100$: CMPB    $CLSBK,#IE.EOF ; EOF ?                     ;[TD001]
394 000330 001402   BEQ     110$          ; IF YES - BRANCH          ;[TD001]
395          ;                               ;[TD001]
396 000332          110$: CALL    $CLDEQ          ; CLOSE CONFIG FILE         ;[TD001]
397 000336 000241   CLC                     ; INDICATE SUCCESS          ;[TD001]
398 000340          RETURN                    ;                               ;[TD001]
399          ;                               ;[TD001]
400          ;                               ;[TD001]
401          ; ERROR CONDITIONS ;[TD001]
402          ;                               ;[TD001]

```

VCFP
STOR

VCFP

VCFP - VNP PSI CONFIGURATION FI MACRO V05.03b Saturday 29-Jun-85 02:17 Page 19-1
STORID - Store id-string according to system type

755 001422

RETURN

: [TD001]

756

: [TD001]

757 001424 000261

; 99\$:

SEC

; Signal string too long

: [TD001]

758 001426

RETURN

: [TD001]

759

: [TD001]

```

52      ;***
53      ; LIBRARY MACROS
54      ;***
55      .MCALL HWDDF$,LBLDF$,PUTRC,DHBD$
56
57      ;***
58      ; LIBRARY SYMBOLS
59      ;***
60      000000      HWDDF$      ; FEATURES MASK BIT DEFINITIONS
61      000000      LBLDF$      ; TASK IMAGE LABEL BLOCK OFFSETS
62      000000      DHBD$      ; DEFINE DECNET HOME BLOCK OFFSETS
63
64      ;***
65      ; LOCAL MACROS
66      ;***
67
68      ; SHIFT LEFT
69      ;
70      .MACRO ASL$ COUNT,REG
71      .IF DF R$SEIS
72      ASH #'COUNT',REG
73      .IFF
74      .REPT COUNT
75      ASL REG
76      .ENDR
77      .ENDC
78      .ENDM ASL$
79
80      ;***
81      ; LOCAL SYMBOLS
82      ;***
83
84      ;
85      ; ERROR CODES RETURNED TO CALLER
86      ;
87      000000      000002      .ASECT
88      .=2      ; 0=SUCCESS
89      ERR2: .BLKW 1      ; OPEN FAILURE
90      ERR4: .BLKW 1      ; GET ATTRIBUTES FAILED
91      ERR6: .BLKW 1      ; FILE NOT CONTIGUOUS
92      ERR8: .BLKW 1      ; LABEL BLOCK READ FAILURE
93      ERR10: .BLKW 1      ; INVALID PROCESS IMAGE
94      ERR12: .BLKW 1      ; PROCESS NOT BUILT FOR APR5 (MAPPED)
95      ERR14: .BLKW 1      ; PROCESS > 4K WORDS (MAPPED)
96      ERR16: .BLKW 1      ; PROCESS EXTENSION > 4K WORDS (MAPPED)
97      ERR18: .BLKW 1      ; PARTITION NOT IN SYSTEM
98      ERR20: .BLKW 1      ; PROCESS/PARTITION BASE MISMATCH
99      ERR22: .BLKW 1      ; PARTITION TOO SMALL
100      ERR24: .BLKW 1      ; NOT COMMON PARTITION
101      ERR26: .BLKW 1      ; PARTITION BUSY
102      ERR28: .BLKW 1      ; CEX PARTITION NOT IN EXEC SPACE (MAPPED)
103      ERR30: .BLKW 1      ; SUB-PCB ALLOCATION FAILURE (MAPPED)
104      ERR32: .BLKW 1      ; MAIN PARTITION TOO FRAGMENTED (MAPPED)
105      ERR34: .BLKW 1      ; PROCESS READ FAILURE
106      ERR36: .BLKW 1      ; CETAB ALLOCATION FAILURE
107      .PSECT

```

VCLQR
MACRO

VCLQRM - REMOVE ENTRIES FROM CL MACRO V05.03b Saturday 29-Jun-85 02:18 Page 5

MACRO DEFINITIONS

54
55
56
57

.SBTTL MACRO DEFINITIONS

.MCALL CLKDF\$,GETRV,PUTRC,VNPDF\$

58 000000
59 000000

CLKDF\$
VNPDF\$

; DEFINE CLOCK QUEUE ENTRY OFFSETS
; DEFINE REQUEST BLOCK OFFSETS AND SYMBOLS

VDEV
Table

VCLQRM - REMOVE ENTRIES FROM CL MACRO V05.03b Saturday 29-Jun-85 02:18 Page 6

VDEV

VDEV - VNP LOAD DEVICE STRUCTUR MACRO V05.03b Tuesday 03-Sep-85 10:28 J 12
Table of contents

5-	54	MACRO DEFINITIONS
6-	64	LOCAL DATA
7-	144	ERROR MESSAGES
8-	188	\$NLDEV - SETUP DATA STRUCTURES
9-	264	FIND - FIND DATA STRUCTURES
10-	306	DVSYM - DEFINE SPECIAL SYMBOLS
11-	375	DBLBL - READ LABEL BLOCK
12-	420	DVALL - ALLOCATE DEVICE TABLE
13-	465	DVXFR - TRANSFER TABLES INTO CORE
14-	639	DVERR - PRINT ERROR
15-	674	DVDEA - DEALLOCATE DATA STRUCTURES
16-	696	REP4A - REPLACE XXXTAB NAME
16-	697	REP4A1 - REPLACE XXX NAME
17-	741	REP9 - REPLACE DEVICE NAME

VDEV
DVXFR


```

465 .SBTTL DVXFR - TRANSFER TABLES INTO CORE
466
467 * DVXFR - TRANSFER DEVICE TABLES INTO CORE
468
469 THIS ROUTINE IS CALLED TO TRANSFER THE DEVICE TABLES INTO CORE, SETUP THE
470 INTRA-BLOCK POINTERS, AND LINK THE DCB INTO THE DEVICE LIST ($DEVHD). THE
471 LOCATIONS CHANGED IN THE DEVICE TABLE IMAGE ARE:
472 D.UCB DCB - POINTER TO UCB
473 D.NAM DCB - DEVICE NAME
474 U.DCB UCB - POINTER TO DCB
475 U.RED UCB - REDIRECT POINTER
476 U.SCB UCB - POINTER TO SCB
477 S.LHD SCB - I/O PACKET LISTHEAD (2 WORDS)
478
479 INPUTS:
480 $LBUF=CORE POOL BUFFER ADDRESS
481 $LLEN=DEVICE TABLES LENGTH
482 DCB=DCB ADDRESS (REAL CORE ADDRESS)
483
484 OUTPUTS:
485 C-BIT=SUCCESS/FAILURE
486
487 -
488
489 DVXFR:
490 001116 016767 000176' 000000G MOV PLEN,$LLEN ; SET TRANSFER SIZE
491 001124 012700 000204' MOV #DEVBUF,R0 ; POINT AT OVERWORKED BUFFER
492 001130 CALL $READ ; TRANSFER DEVICE TABLES INTO BUFFER
493 001134 103002 BCC 5$ ; ALL OK
494 001136 000167 000544 JMP 101$ ; FAILED TO READ IN DEVICE DATA STRUCTURES
495
496 001142 016767 000174' 000200' 5$ MOV PBUF,VREL ; COMPUTE RELOCATION DIFFERENTIAL
497 001150 160067 000200' SUB R0,VREL ; FOR DEVBUF
498
499 001154 016700 000150' MOV DCB,R0 ; GET DCB ADDRESS
500 001160 166700 000200' SUB VREL,R0 ; SUBTRACT DIFFERENTIAL
501 001164 016760 000000G 000000G MOV $$DEV,D.NAM(R0) ; SETUP DEVICE NAME
502 001172 005001 CLR R1 ; GET FIRST UNIT NUMBER
503 001174 156001 000000G BJSB D.UNIT(R0),R1 ; AND LAST UNIT NUMBER
504 001200 005003 CLR R3
505 001202 156003 000001G BJSB D.UNIT+1(R0),R3 ; CALCULATE NUMBER OF UNITS
506 001206 160103 SUB R1,R3
507 001210 005203 INC R3
508 001212 003002 BGT 7$ ; ALL OK
509 001214 000167 000502 JMP 111$ ; IF NOT POSITIVE, ILLEGAL
510
511 001220 010367 000172' 7$ MOV R3,UNITS ; SAVE IT FOR LATER
512 001224 010002 MOV R0,R2 ; COPY DCB ADDRESS
513 001226 062702 000000G ADD #D.UCB,R2 ; GET D.UCB ADDRESS
514 001232 CALL VAD ; VALIDATE IT
515 001236 103002 BCC 8$ ; BR IF LEGAL
516 001240 000167 000464 JMP 121$ ; IF CS, ILLEGAL
517 001244 011202 8$ MOV (R2),R2 ; GET FIRST UCB ADDRESS
518 001252 103002 CALL RVAD ; RELOCATE AND VALIDATE IT
519 001254 000167 000456 BCC 9$ ; BR IF LEGAL
520 001260 010246 9$ JMP 131$ ; IF CS, ILLEGAL
521 001262 066716 000200' MOV R2,-(SP) ; COMPUTE TRUE ADDRESS
522 ADD VREL,(SP)

```

VDEV CREATED BY MACRO ON 3-SEP-85 AT 10:28 PAGE 4 I 14

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
CALL	8-211 8-213 8-217 8-219 8-221 8-223 8-226 8-232 8-233 8-250 9-281 10-326 10-331 10-343 10-345 10-352 10-360 10-364 11-393 11-394 11-400 12-443 13-491 13-513 13-517 13-525 13-529 13-558 13-569 13-587 13-596 13-598 13-600 13-602 14-650 14-654 14-667 14-670 15-690 15-693 16-709 16-718 16-731 16-733
CALLR	10-373
EMSG\$	#5-58 10-352 10-360 10-364
EMSG\$R	#5-58 8-262 9-303 9-304 11-414 11-415 11-417 11-418 12-463 13-609 13-610 13-611 13-614 13-615 13-616 13-619 13-620
ERBLK\$	#8-208
ERMSG\$	#8-208
ERRPT\$	#8-208
GETAD	#5-58
GETRV	#5-58 8-226 8-233 9-281 13-587 13-596 13-600
NTLR\$	#5-58 7-148 7-149 7-150 7-151 7-152 7-153 7-154 7-155 7-156 7-157 7-158 7-159 7-160 7-161 7-162 7-163 7-164 7-165 7-166 7-167 7-168 7-169
PUTAD	#5-58
PUTRC	#5-58 8-232 8-250 13-569 13-598 13-602
RAND	#8-226 8-226 #8-232 8-232 #8-233 8-233 #8-250 8-250 #9-281 9-281 #13-569 13-569 #13-587 13-587 #13-596 13-596 #13-598 13-598 #13-600 13-600 #13-602 13-602
RETURN	8-256 9-298 10-367 11-409 12-458 13-603 13-637 14-672 15-694 16-739 17-756
SWTCH1	#5-59 8-210 8-255 12-442 12-446 15-688 15-691
SWTCHO	#5-59 8-209 8-254 12-441 12-445 13-575 13-577 15-689 15-692
UCBDF\$	#5-58 5-62
VNPBPT	#8-207 8-208

VDIS - VNP SHOW COMMANDS
\$DOBJ - SHO OBJ

MACRO V05.03b Monday 15-Jul-85 12:14¹⁵ Page 9-1

424 002434
425 002442
426
427 002452
428 002460
429
430
431
432
433
434 002462
435 002506
436
437

```
60$: DIR$ #TIDET           ; DETACH TERMINAL
     ERRPT$ OERR1,$EROUT   ; DISPLAY OBJECT NOT FOUND MESSAGE

70$: DIR$ #TIDET           ; DETACH TERMINAL
     RETURN

     .ENABL LC

;
; ERROR MESSAGES
;
     ERMSG$ <Object not in system>
     ERBLK$ OERR1

     .DSABL LC
```

VDIS - VNP SHOW COMMANDS MACRO V05.03b Monday 15-Jul-85 12:14 Page 16
 CVTDEV - CONVERT DEVICE NAME TO ASCII

```

933                                     ,SBTTL  CVTDEV - CONVERT DEVICE NAME TO ASCII
934
935                                     :+
936                                     :
937                                     : CVTDEV - CONVERT DEVICE NAME TO ASCII
938                                     :
939                                     : INPUTS:
940                                     : R0 - NEXT AVAILABLE BYTE IN OUTPUT BUFFER
941                                     : R3 - POINTER TO TWO WORD DEVICE NAME
942                                     :       WORD 1 - TWO CHARACTER ASCII NAME
943                                     :       WORD 2 - BINARY UNIT NUMBER
944                                     :
945                                     : OUTPUTS:
946                                     : R0 - NEXT AVAILABLE BYTE IN OUTPUT BUFFER
947                                     : R1,R2 DESTROYED
948                                     :
949                                     :-
950 005214 112320 CVTDEV: MOVB  (R3)+,(R0)+      ; STORE FIRST HALF OF DEVICE NAME
951 005216 112320      MOVB  (R3)+,(R0)+      ; STORE SECOND HALF OF NAME
952 005220 012301      MOV   (R3)+,R1        ; GET DEVICE UNIT NUMBER
953 005222 005002      CLR   R2              ; SUPPRESS ZEROES
954 005224      CALL  $CBOMG                ; CONVERT IT TO ASCII
955 005230 112720 000072      MOVB  #':,(R0)+    ; STORE TRAILING COLON
956 005234      RETURN

```

332
333
334
335
336
337
338
339
340
341
342
343

```

854
855
856      ; MSGOUT - OUTPUT MESSAGE
857
858      ; INPUTS:
859      ;       R2 = ARGUMENT BLOCK ADDRESS FOR $ERMSG
860
861      ; OUTPUTS:
862      ;       TYPE MESSAGE
863
864      MSGOUT: CALL    $SAVAL          ; SAVE REG'S
865      002074 012700 000254'      MOV    #MSGB,R0      ; STORE BUFFER ADDRESS
866      002100      CALL    $ERMSG      ; FORMAT MESSAGE
867      002104      CALL    $TIOUT      ; TYPE MESSAGE
868      002110      RETURN
869
870
871      000001      .END

```

```

55 .SBTTL MACRO DEFINITIONS
56 :***
57 : LIBRARY MACROS
58 :***
59 .MCALL ASL$,EMSG$R,ISTAT$,NTLRS$,STATS$,TRANS$,CHADF$,PUTRC
60 .MCALL SAYRG,RESRG,GETAD,GETRV,CEACC$,XPDDB$,PLBDBF$,DHBDF$
61 .ENABL LC
62
63 000000 XPDDB$ : DEFINE XPT OFFSETS
64 000000 PLBDBF$ : DEFINE PLB OFFSETS
65 000000 CHADF$ : DEFINE CHARACTERISTICS WORD OFFSETS
66 000000 DHBDF$ : DEFINE DECNET HOME BLOCK OFFSETS
67
68
69
70 :***
71 : LOCAL MACROS
72 :***
73
74 :
75 : REJECT TPARS TRANSITION
76 :
77 .MACRO REJ$
78 ADD #2,(SP) : RETURN TO CALLER+2
79 CLR SYNERR : INDICATE NO SYNTAX ERROR
80 .ENDM REJ$

```

```

593      .SBTTL STORID - Store id-string according to system type
594
595      *** STORID - Store id-string according to system type
596
597      FUNCTION
598
599          This routine is used to check and store various parameters
600          during the parsing of CETAB macros. In PSI V2.0 compatible
601          systems, it checks for and stores parameters in PSI V2.0
602          compatible format. In extended systems, id-strings of up
603          to MAXID characters are permitted and stored appropriately.
604          The buffer is padded with spaces. The length returned in the
605          buffer is the maximum length for old-format systems (fixed
606          fields) or the actual length for new-format systems (variable
607          fields).
608
609      INPUTS
610
611          R0 - address of buffer to contain string prefixed by length
612          R1 - maximum length of string in old-format system
613              (R1 < MAXID)
614          .PSTCN - actual length of string
615          $PFLAG - PSI system type flags
616
617      OUTPUTS
618
619          R0, R1, R2 destroyed
620          String and length stored in buffer
621          Carry set if invalid string length for system type
622
623      -
624      STORID::
625          MOV R0, -(SP) ; Save buffer address
626          MOV R1, -(SP) ; Save maximum length
627          CLR (R0)+ ; Clear length field in output
628          MOV #MAXID, R1 ; Always clear MAXID spaces
629          CALL SPFILL
630          MOV (SP)+, R1 ; Restore length
631          MOV (SP)+, R2 ; Get back buffer address
632          BIT #PF.XDF, $PFLAG ; Check for extended data format
633          BNE 20$
634
635          CMP .PSTCN, R1 ; Old format, so use max length in R1
636          BHI 99$ ; HI, then string too long
637          MOVB R1, (R2)+ ; Set length to maximum for fixed field
638          BR 30$
639
640          CMP .PSTCN, #MAXID ; New format, max length is MAXID
641          BHI 99$
642          MOVB .PSTCN, (R2)+ ; Set actual length for variable field
643
644          ; Move string to buffer
645
646          30$: MOV .PSTCN, R1
647              MOV .PSTPT, R0 ; Get address of source
648              CALL MOVE ; Move string to buffer
649              CLC ; Signal success
    
```

1015
 1016
 1017
 1018
 1019
 1020
 1021
 1022
 1023
 1024
 1025
 1026
 1027
 1028
 1029
 1030
 1031
 1032
 1033
 1034
 1035
 1036
 1037
 1038
 1039
 1040
 1041
 1042
 1043
 1044
 1045
 1046
 1047
 1048
 1049
 1050
 1051
 1052
 1053
 1054
 1055
 1056
 1057
 1058
 1059
 1060
 1061
 1062
 1063


```

;SBTTL LLC$DF STATE TABLE
;
; LLC DEFINITION (LLC$DF)
;
STATES$ LLCDF
TRANS$ "LLC",,,,1,SYNERR
STATES$
TRANS$ '$
STATES$
TRANS$ 'DF'
STATES$ ; LLC NAME
TRANS$ $RAD50,,L.NAME
STATES$
TRANS$ '<,>'
STATES$ ; FLAGS
TRANS$ !BITS
STATES$ ; PRIORITY
TRANS$ $NUMBR
STATES$
TRANS$ '<,>'
STATES$ ; EXTRA PDV CHANNELS
TRANS$ $NUMBR
STATES$
TRANS$ !END,$EXIT
TRANS$ '<,>'
STATES$ LLCDF1 ; 2 CHARACTER DEVICE NAME
TRANS$ '<,>',INC
TRANS$ $ALPHA,,L.CH1
STATES$
TRANS$ $ALPHA,,L.CH2
STATES$
TRANS$ !END,$EXIT
TRANS$ '<,>'
STATES$ INC ; LINE TABLE COUNT
TRANS$ $NUMBR,EXT,P.INC
TRANS$ $LAMDA
STATES$ EXT
TRANS$ !END,$EXIT
TRANS$ '<,>'
STATES$ ; LLC PROCESS EXTENSION
TRANS$ '<,>',LCTIM
TRANS$ $NUMBR,$EXIT,P.EXT
STATES$ LCTIM
TRANS$ $NUMBR,,S.CTIM ; COUNTER TIMER
STATES$
TRANS$ !END,$EXIT
STATES$

```

```

1567 .SBTTL FEASDF ACTION ROUTINES
1568 ;
1569 ; FIRST CHARACTER OF DEVICE NAME (FEASDF)
1570 ;
1571 003734 126767 000000G 000000G F.CH1: CMPB .PCHAR,$DEV ; IS THIS THE RIGHT FEATURES DEFINITION ?
1572 003742 001404 BEQ 10$ ; IF EQ, YES
1573 003744 REJ$ ; ELSE, REJECT TRANSITION
1574 003754 10$: RETURN
1575 ;
1576 ; SECOND CHARACTER OF DEVICE NAME (FEASDF)
1577 ;
1578 003756 126767 000000G 000001G F.CH2: CMPB .PCHAR,$DEV+1 ; ARE WE SURE THIS IS THE RIGHT ONE ?
1579 003764 001404 BEQ 10$ ; IF EQ, YES
1580 003766 REJ$ ; ELSE, REJECT TRANSITION
1581 003776 10$: RETURN
1582 ;
1583 ; INITIAL FEATURES WORD (FEASDF)
1584 ;
1585 004000 016767 000000G 000000G F.FEA: MOV .PNUMB,$FEATR ; SETUP THE FEATURES WORD
1586 004006 RETURN
1587 ;
1588 ; MODIFIABLE BITS IN THE FEATURES WORD (FEASDF)
1589 ;
1590 004010 046767 000000G 000000G F.MOD: BIC .PNUMB,$FEATR ; CLEAR THE MODIFIABLE BITS
1591 004016 RETURN
1592 ;
1593 ; BITS CURRENTLY SET IN THE FEATURES WORD (FEASDF)
1594 ;
1595 004020 056767 000000G 000000G F.SET: BIS .PNUMB,$FEATR ; SET THE CURRENT STATES
1596 004026 RETURN

```

VCFG
MACRO
MACRO
ASL\$
CALL

CALLR
CEACCS
CHADFS
DBGTPS

DHBD\$
EMSGSR

ERBLK\$
ERMSG\$
ERRPT\$
GETAD
GETRV
I\$STAT\$
M\$TRAN\$
NTL\$R\$

PLBDF\$
PUTRC
RAND
REJ\$
RESRG
RETURN

SAVRG
SOB
STATES

MACRO NAME	REFERENCES									
ASL\$	#5-59	27-1332								
CALL	8-261	8-268	8-272	8-283	8-286	8-288	8-291	8-293	8-296	8-298
	8-300	8-304	8-318	8-320	8-322	8-328	8-330	8-334	8-336	8-342
	8-344	8-350	8-352	8-354	8-356	8-359	8-361	8-367	8-369	8-375
	8-377	8-383	8-385	8-391	8-393	8-397	8-402	8-404	8-412	8-414
	8-428	9-458	9-459	10-482	11-540	12-567	12-572	12-585	12-588	13-629
	13-648	27-1299	28-1395	28-1437	29-1476	29-1479	33-1612	33-1616	33-1617	33-1621
	16-711									
CALLR	#5-60	33-1616								
CEACC\$	#5-59	5-65								
CHADF\$	#17-717	#17-747	#17-750	#17-753	#17-756	#17-759	#17-762	#17-763	#17-766	#17-780
DBGTP\$	#17-781	#17-782	#17-785	#17-786	#17-789	#17-790	#18-810	#18-814	#18-821	#18-828
	#20-859	#20-875	#20-882	#22-946	#22-949	#22-953	#22-957	#22-961	#22-965	#22-969
	#22-982	#24-1039	#24-1042	#24-1047	#24-1050	#24-1053	#24-1056			
DHBDP\$	#5-60	5-66								
EMSGR	#5-59	8-363	8-435	8-436	8-437	8-438	8-439	8-440	8-441	8-442
	8-443	9-467	10-497	10-498	11-550	12-1123	12-1169	12-1181	12-1192	12-1207
	25-1226	25-1227	25-1246	25-1258	26-1274	27-1291	27-1292	27-1304	27-1336	28-1363
	28-1364	28-1387	28-1442	28-1443	30-1514	30-1524	31-1542	31-1552		
ERBLK\$	#8-260									
ERMSG\$	#8-260									
ERRPT\$	#8-260									
GETAD	#5-60	33-1617								
GETRV	#5-60	33-1612								
ISTAT\$	#5-59	17-717								
MTRANS	#17-717									
NTLRS	#5-59	7-180	7-181	7-182	7-183	7-184	7-185	7-186	7-187	7-188
	7-189	7-190	7-191	7-192	7-193	7-194	7-195	7-196	7-197	7-198
	7-199	7-200	7-201	7-202	7-203	7-204	7-205	7-206	7-207	7-208
	7-209	7-210	7-211	7-212	7-213	7-214	7-215	7-216	7-217	7-218
	5-64									
PLBDF\$	#5-60									
PUTRC	#5-59									
RAND	#33-1612	33-1612	#33-1617	33-1617						
REJS	#5-77	25-1087	25-1131	25-1139	25-1158	32-1573	32-1580			
RESRG	#5-60	29-1500								
RETURN	8-430	9-462	10-487	11-544	12-591	13-650	13-653	14-675	15-697	25-1088
	25-1099	25-1110	25-1121	25-1132	25-1140	25-1148	25-1156	25-1159	25-1168	25-1180
	25-1191	25-1205	25-1225	25-1244	25-1256	26-1272	27-1290	27-1303	27-1319	27-1335
	28-1362	28-1386	28-1403	28-1417	28-1441	29-1501	30-1513	30-1523	30-1531	31-1540
	31-1550	31-1559	31-1565	32-1574	32-1581	32-1586	32-1591	32-1596	33-1622	34-1654
	35-1669	35-1682								
SAVRG	#5-60	29-1451								
SOB	34-1647									
STATE\$	#5-59	17-723	#17-725	#17-727	#17-729	#17-731	#17-733	#17-735	#17-737	#17-739
	#17-741	#17-743	#17-745	#17-748	#17-751	#17-754	#17-757	#17-760	#17-764	#17-767
	#17-769	#17-771	#17-778	#17-783	#17-787	#18-796	#18-799	#18-802	#18-805	#18-808
	#18-812	#18-816	#18-819	#18-823	#18-826	#18-830	#19-839	#19-841	#19-847	#19-849
	20-857	#20-861	#20-864	#20-867	#20-870	#20-873	#20-877	#20-880	#20-884	21-891
	#21-893	#21-895	#21-897	#21-899	#21-901	#21-903	#21-907	#22-932	#22-934	#22-936
	#22-938	#22-940	#22-942	#22-944	#22-947	#22-951	#22-955	#22-959	#22-963	#22-967
	#22-971	#22-974	#22-980	#23-988	#23-991	#23-994	#23-997	#23-1000	#23-1003	#23-1006
	#23-1009	#23-1012	24-1019	#24-1021	#24-1023	#24-1025	#24-1027	#24-1029	#24-1031	#24-1033

J 8
VCFP - VNP PSI CONFIGURATION FI MACRO V05.03b Saturday 29-Jun-85 02:17 Page 9-2
\$NTCFP - CONFIG FILE SCAN

403 000342
404 000350
405 000356
406 000364
407 000372

101\$: EMSG\$R M4
102\$: EMSG\$R M5
103\$: EMSG\$R M6
104\$: EMSG\$R MA
105\$: EMSG\$R MB

; X3P\$DF MISSING
; DSA\$DF MISSING
; DSC\$DF MISSING
; DSN\$DF missing
; RNA\$DF missing

;[TD001]
;[TD001]

VCFP
MOVE

```

761                                     .SBTTL MOVE - Move bytes                :[TD001]
762                                     :+                                     :[TD001]
763                                     *** MOVE - Move bytes                    :[TD001]
764                                     :                                     :[TD001]
765                                     INPUTS                                   :[TD001]
766                                     R0 - Source address                      :[TD001]
767                                     R1 - Number of bytes to move             :[TD001]
768                                     R2 - Target address                      :[TD001]
769                                     :                                     :[TD001]
770                                     OUTPUTS                                  :[TD001]
771                                     R2 - Address of last byte moved + 1       :[TD001]
772                                     R0, R1 destroyed                        :[TD001]
773                                     :                                     :[TD001]
774                                     :-                                     :[TD001]
775 001430 MOVE:: TST R1 ; Check for nothing to do                          :[TD001]
776 001430 BEQ 20$                                     :[TD001]
777 001432 005701                                     :[TD001]
778 001434 112022 10$: MOVB (R0)+,(R2)+                :[TD001]
779 001436 005301 DEC R1                               :[TD001]
780 001440 003375 BGT 10$                             :[TD001]
781 001442 20$: RETURN                                :[TD001]

```

```
109          ;***  
110          ; LOCAL DATA  
111          ;***  
112  
113          .PSECT DATA,D  
114  
115          ;  
116          ; ERROR INDICATION  
117          ;  
118          ERROR: .BLKW 1  
119          ;  
120          .PSECT .LBUF,D  
121  
122          ;  
123          ; DISK BUFFER FOR LABEL BLOCK READ  
124          ;  
125          DSKBUF: .BLKW 1  
126          ;  
127          .PSECT
```

VCLOR
LOCAL

VCLORM - REMOVE ENTRIES FROM CL MACRO V05.03b Saturday 29-Jun-85 02:18 ^{J 11} Page 6
LOCAL DATA

.SBTTL LOCAL DATA

61
62
63
64
65
66
67

⋮ LOCAL DATA
⋮

000000

AUXPDV: .BLKW 1

; AUX PDV ADDRESS

VDEV

.TITLE VDEV - VNP LOAD DEVICE STRUCTURES
.IDENT /V05.00/

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - RSX11M/S DEVICE DATA STRUCTURE ROUTINES (NS: AND NX:)

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 27-FEB-78
VERSION 2.0 RELEASE
- 2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
- 3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
- 4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/RSX V1.0


```

522 001266 012660 000000G      MOV      (SP)+,D.UCB(R0) ; SET FIRST UCB ADDRESS
523 001272 010201      10$: MOV      R2,R1      ; PUT UCB ADDRESS IN R1
524 001274 062702 000020      ADD      #U.SCB,R2      ; GET U.SCB ADDRESS
525 001300      CALL      VAD      ; VALIDATE IT
526 001304 103002      BCC      11$      ; BR IF LEGAL
527 001306 000167 000444      JMP      141$      ; IF CS, ILLEGAL
528 001312 011202      11$: MOV      (R2),R2      ; GET SCB ADDRESS
529 001314      CALL      RVAD      ; RELOCATE AND VALIDATE IT
530 001320 103002      BCC      12$      ; BR IF LEGAL
531 001322 000167 000436      JMP      151$      ; IF CS, ILLEGAL
532 001326 010046      12$: MOV      R0,-(SP)      ; COMPUTE TRUE
533 001330 066716 000200'      ADD      VREL,(SP)      ; ADDRESS
534 001334 012661 000000      MOV      (SP)+,U.DCB(R1) ; SETUP DCB ADDRESS
535 001340 010146      MOV      R1,-(SP)      ; COMPUTE TRUE
536 001342 066716 000200'      ADD      VREL,(SP)      ; ADDRESS
537 001346 012661 000002      MOV      (SP)+,U.RED(R1) ; REDIRECT POINTER
538 001352 010246      MOV      R2,-(SP)      ; COMPUTE TRUE
539 001354 066716 000200'      ADD      VREL,(SP)      ; ADDRESS
540 001360 012661 000020      MOV      (SP)+,U.SCB(R1) ; SCB ADDRESS
541 001364 062702 000000G      ADD      #S.LHD,R2      ; AND I/O PACKET LISTHEAD
542 001370 005012      CLR      (R2)
543 001372 010246      MOV      R2,-(SP)      ; COMPUTE TRUE
544 001374 066716 000200'      ADD      VREL,(SP)      ; ADDRESS
545 001400 012662 000002      MOV      (SP)+,2(R2)      ; ...
546
547      .IF DF R$MPL ;[MP01]
548      SUB      #S.LHD,R2 ; GET SCB ADDRESS ;[MP01]
549      ADD      .SK$5,R2 ; POINT TO S.K$5 ;[MP01]
550      MOV      $BIAS,(R2) ; SET UP BIAS FOR CON ONLI ;[MP01]
551
552      .ENDC ;[MP01]
553
554 001404 005303      DEC      R3      ; DONE?
555 001406 001407      BEQ      20$      ; IF EQ, YES
556 001410 010102      MOV      R1,R2      ; COPY UCB ADDRESS
557 001412 066002 000000G      ADD      D.UCBL(R0),R2 ; ADD UCB LENGTH FROM DCB
558 001416      CALL      VAD      ; VALIDATE NEXT UCB ADDRESS
559 001422 103323      BCC      10$      ; BR IF OK - SETUP NEXT UCB/SCB PAIR
560 001424 000544      BR      131$      ; ELSE ILLEGAL
561
562 001426 016746 000176'      20$: MOV      PLEN,-(SP)      ; SAVE LENGTH OF DATA STRUCTURES
563 001432 016746 000174'      MOV      PBUF,-(SP)      ; AND ADDRESS OF DATA STRUCTURES
564 001436 012746 000204'      MOV      #DEVBUF,-(SP) ; GET BUFFER ADDRESS
565 001442 016767 000176' 000000G 25$: MOV      PLEN,$RSIZE ; ASSUME MAXIMUM RECORD SIZE NOT EXCEEDED
566 001450 022767 001000 000176'      CMP      #1000,PLEN ; MAXIMUM RECORD SIZE EXCEEDED?
567 001456 103003      BHIS      27$      ; BR IF NO
568 001460 012767 001000 000000G      MOV      #1000,$RSIZE ; ELSE ONLY WRITE 512. BYTE RECORD
569 001466      PUTRC      PBUF      ; WRITE DATA STRUCTURES
570 001500 022767 001000 000176'      CMP      #1000,PLEN ; MORE TO OUTPUT?
571 001506 103013      BHIS      28$      ; BR IF NO
572 001510 162767 001000 000176'      SUB      #1000,PLEN ; UPDATE DATA STRUCTURE LENGTH
573 001516 062767 001000 000174'      ADD      #1000,PBUF ; UPDATE ADDRESS
574 001524 062716 001000      ADD      #1000,(SP) ; UPDATE BUFFER ADDRESS
575 001530      SWTCHO      (SP) ; USE NEW BUFFER ADDRESS
576 001534      BR      25$      ; AND WRITE NEXT RECORD
577 001536      SWTCHO      (SP)+ ; RETURN TO DEFAULT BUFFER
578 001542 012667 000174'      28$: MOV      (SP)+,PBUF ; ELSE RESTORE ADDRESS

```

[illegible]

```

LL          SSSSSSSS  TTTT TTTT TTT
LL          SSSSSSSS  TTTT TTTT TTT
          SS          TT
LL          SS          TT
LL          SS          TT
LL          SS          TT
LL          SSSSSS     TT
LL          SSSSSS     TT
          SS          TT
LL          SS          TT
LL          SS          TT
LL          SS          TT
LL          SS          TT
LLLLLLLLLL SSSSSSSS   TT
LLLLLLLLLL SSSSSSSS   TT

```

```

439      .SBTTL $DALI - SHOW ALIAS
440
441      +
442      :
443      : $DALI - SHOW ALL ALIASES
444      : SHOW KNOW ALIASES
445      : SHO ALIAS X
446      :
447      : INPUTS: NONE
448      :
449      : OUTPUTS: ALIAS INFORMATION IS DISPLAYED ON THE USER'S TERMINAL
450      :
451      : ALL REGISTERS ARE DESTROYED
452      :
453      : -
454
455      002512      $DALI:: CALL      CKTYPE      : GET COMMAND TYPE
456      002516      103004      BCC      10$      : BR IF SUCCESS
457      002520      ERRPT$      SYNERR,$EROUT    : ELSE SYNTAX ERROR
458      002530      012701      001062'      10$: MOV      #TEMP2,R1      : STORAGE FOR HEADER STRING
459
460      002534      SWITCHI      #$HMBUF      : USE TEMPORARY BUFFER
461      002542      GETRV      $DECPT,#D$END      : READ DECNET HOME BLOCK
462      002562      SWITCHI      : RESET TO DEFAULT INPUT BUFFER
463
464      002570      012703      000000G      MOV      #$HMBUF+$ANN,R3      : ALIAS NAME BLOCK LISTHEAD
465      002574      012705      000100      MOV      #100,R5      : LENGTH OF ALIAS NAME BLOCK
466      002600      005767      000000G      TST      $QUAL      : SHOW SPECIFIC ALIAS?
467      002604      001422      BEQ      40$      : BR IF YES
468      002606      005713      TST      (R3)      : ANY ALIASES IN SYSTEM?
469      002610      001533      BEQ      150$      : BR IF NO
470      002612      022767      000004      000000G      CMP      #0$KNO,$QUAL      : SHOW KNOWN ALIASES?
471      002620      001464      BEQ      100$      : BR IF YES
472
473      : SHOW ALL ALIASES
474      :
475      002622      012700      000135'      MOV      #AALL,R0      : MUST BE SHOW ALL ALIASES
476      002626      112021      20$: MOVVB      (R0)+,(R1)+      : STORE HEADER STRING
477      002630      001376      BNE      20$      :
478      002632      005004      CLR      R4      : INDICATE HEADER IS TO BE DISPLAYED
479      002634      30$: CALL      NXTBLK      : GET NEXT ALIAS NAME BLOCK IN LIST
480      002640      103517      BCS      150$      : BR IF AT END OF LIST
481      002642      CALL      DISALI      : DISPLAY ALIAS INFORMATION
482      002646      005204      INC      R4      : INDICATE HEADER NOT TO BE DISPLAYED
483      002650      000771      BR      30$      : GET NEXT BLOCK
484
485      : SHOW SPECIFIC ALIAS
486      :
487      002652      005767      000000G      40$: TST      $OPTION      : WAS SCOPE SPECIFIED?
488      002656      001501      BEQ      140$      : BR IF NO
489      002660      005703      TST      R3      : ANY ALIASES IN SYSTEM?
490      002662      001470      BEQ      130$      : BR IF NO
491      002664      012700      000167'      MOV      #ASPEC,R0      : HEADER FOR SPECIFIC ALIAS
492      002670      112021      50$: MOVVB      (R0)+,(R1)+      : STORE IT
493      002672      001376      BNE      50$      :
494      002674      005004      CLR      R4      : INDICATE HEADER TO BE PRINTED
495      002676      012700      000000G      60$: MOV      #RGB,R0      : GET INPUT PARAMETERS ADDRESS

```

```

958                                     .SBTTL CHKSNK - CHECK FOR MATCH ON SINK TYPE
959
960                                     +
961                                     :
962                                     : CHKSNK - CHECK FOR MATCH ON SINK TYPE
963                                     :
964                                     : INPUTS:
965                                     : R3 - ADDRESS OF FILTER BLOCK
966                                     : TEMP3 - NODE ADDRESS OF LAST SINK NODE DISPLAYED
967                                     : LO.SKA+RQB - SPECIFIED SINK NODE
968
969                                     : OUTPUTS:
970                                     : CARRY CLEAR:
971                                     : MATCH ON SINK NODE
972
973                                     : CARRY SET:
974                                     : NO MATCH
975
976                                     -
977 005236 032767 002000 000000G CHKSNK: BIT #OP$KSK,$OPTON : DISPLAY KNOWN SINKS?
978 005244 001014 BNE 20$ : BR IF YES - ALWAYS MATCH
979 005246 032763 100000 000014 BIT #FF.REM,F.FLG(R3) : EVENT QUALIFIED BY SINK NODE?
980 005254 001004 BNE 10$ : BR IF YES
981 005256 005767 000014G TST RQB+LO.SKA : IS SINK NODE EXECUTOR?
982 005262 001034 BNE 50$ : BR IF NO
983 005264 000404 BR 20$ : CHECK TO DISPLAY SINK NODE
984 005266 026367 000026 000014G 10$: CMP F.REM(R3),RQB+LO.SKA : DO SINK NODES MATCH?
985 005274 001027 BNE 50$ : BR IF NO
986 005276 032763 100000 000014 20$: BIT #FF.REM,F.FLG(R3) : IS FILTER QUALIFIED BY SINK NODE?
987 005304 001010 BNE 30$ : BR IF YES
988 005306 005767 173670 TST TEMP3 : IS CURRENT SINK NODE EXECUTOR?
989 005312 001416 BEQ 40$ : BR IF YES - MATCH
990 005314 005067 173662 CLR TEMP3 : ELSE BUILD SINK NODE MESSAGE
991 005320 CALL DLOS NK : BUILD SINK NODE MESSAGE
992 005324 000414 BR 60$ : AND EXIT
993 005326 026367 000026 173646 30$: CMP F.REM(R3),TEMP3 : HAS SINK NODE BEEN DISPLAYED ALREADY?
994 005334 001405 BEQ 40$ : BR IF YES
995 005336 016367 000026 173636 MOV F.REM(R3),TEMP3 : ELSE BUILD SINK NODE MESSAGE
996 005344 CALL DLOS NK : DISPLAY SINK NODE
997 005350 000241 40$: CLC : INDICATE SUCCESS
998 005352 000401 BR 60$
999 005354 000261 50$: SEC : INDICATE NO MATCH
1000 005356 60$: RETURN

```

```

345
346      ; TRANSFER CETAB INTO DISK IMAGE
347
348 000242 052767 000000G 000000G BIS #FE.CEX,$Fmask ; GET ALLOCATION FROM CEX POOL
349 000250 016767 000000G 000000G MOV .CEPWR,$BFR ; GET ADDRESS OF COM/EXEC POWERFAIL ROUTINE
350 000256      PUTRC .NETPF,#2 ; STORE ADDRESS OF POWERFAIL ROUTINE
351 000276 012700 000000G      MOV #SDFUI,R0 ; SETUP ASCII FILE SPEC
352 000302 012701 000000G      MOV #SCET,R1 ; FOR CXXX,5XJCETAB.STB
353 000306 005002      CLR R2 ; ...
354 000310      CALL $AZFS ;
355 000314      CALL $CTBIO ; TRANSFER CETAB INTO DISK IMAGE
356 000320 103454      BCS TSKERR ; IF CS, ERROR
357 000322      CALL NLBLK ; SETUP NETLDR DATA BLOCK
358 000326      CALL CEPTB ; RELOCATE CETAB TABLE POINTERS
359 000332 010102      MOV R1,R2 ; COPY CETAB ALLOCATION ADDRESS
360
361      .IF NDF R$SMPL ;[MP01]
362 000334 016767 000000G 000000G MOV $CEAVL,$BFR ; THE COMM EXEC LISTHEAD MIGHT HAVE CHANGED !
363 000342      PUTRC .CEAVL,#2 ; WRITE OUT THE START OF CEX FREE SPACE ;[MP01]
364      .ENDC
365
366 000362      CALL $VIMI ; LOAD COMEXEC IMAGE INTO INTERN. TEMPL.
367 000366 103431      BCS TSKFPR ; IF ERROR - BRANCH
368 000370 016767 000000G 000000G MOV $$MUX,$NTLPT ; STORE POOL ADDRESS
369
370      ;+
371      ; ALLOCATE DECNET HOME BLOCK
372      ; -
373
374 000376      CALL $DECHM ; ALLOCATE DECNET HOME BLOCK
375 000402 103423      BCS TSKERR ; IF ERROR -BRANCH
376
377 000404 012701 000006      MOV #6,R1 ; LENGTH OF NODE NAME
378
379 000410      GETRV $DECPTR,$DSEND ; READ DECNET HOME BLOCK
380
381 000430 012700 000006G      MOV #BFR+$LNAM,R0 ; POINT TO EXECUTOR NODE NAME
382 000434 112720 000040      MOVB #'(R0)+ ; INIT TO SPACES ( MAY BE PSI ONLY SYSTEM)
383 000440      SOB R1,$0$ ; ...
384
385 000444      PUTRC ; WRITE BACK DECNET HOME BLOCK
386
387 000450      RETURN
388
389      ;
390      ; ERROR CONDITIONS
391
392 000452 042767 000000G 000000G TSKERR: BIC #FE.CEX,$Fmask ; SAY CEX NOT LOADED
393 000460 016000 000724      MOV ETAB2-2(R0),R0 ; GET ERROR MESSAGE CONTROL BLOCK ADDRESS
394 000464      CALLR @R0+ ; PRINT ERROR MESSAGE

```

AB.NPV= 000001	A.PCBL 000000	C.CSTP= 000012	D\$SEG 000036	I.XPBF 000022
AB.TYP= 000002	A.PLEN= 000016	C.DST 000016	D\$SER 000032	I.XPBL 000026
AF.AST= 000010	A.POWE 000004	C.EFN 000003	D\$SRL 000052	I.XPBV 000030
AF.ESQ= 000020	A.PRI 000002	C.LGTH= 000020	D\$BUG= 177514	I.XRBF 000012
AF.LCK= 000040	A.PRM 000012	C.LNK 000000	D\$ISK= 000000	I.XRBL 000016
AF.MDE= 000200	A.PROC 000020	C.MRKT= 000000	D\$LL1= 000001	I.XTCB 000004
AF.NOT= 000002	A.RECE 000016	C.PCPL 000011	D\$SYNC= 000000	I.XTMO 000020
AF.OOB= 000004	A.REL 000000	C.PDPL 000010	D\$SYNM= 000000	I.XTTB 000032
AF.QUE= 000100	A.RES = 000032	C.PNAM 000002	ERR1 000034R	002 I.XTTL 000036
AF.XCC= 000001	A.SBUF 000022	C.PRMT 000012	ERR2 000066R	002 K\$CNT= 177546
AK.BUF= 000200	A.SLEN 000024	C.PSTS 000006	ERR3 000152R	002 K\$CSR= 177546
AK.GBI= 000202	A.SMAP 000020	C.PTCB 000000	ERR4 000224R	002 K\$LDC= 000000
AK.GGF= 000303	A.STA 000015	C.ROT 000002	ERR5 000252R	002 K\$TPS= 000074
AK.OCB= 000201	A.STAT 000010	C.RSI 000012	ETAB1 000716R	002 LD\$LP = 000000
AS.CAA= 000006	A.TCB 000004	C.SCHD= 000002	ETAB2 000726R	002 LO.CSN 000006
AS.DEL= 000010	A.TCBL 000006	C.SRC 000014	E\$XPR= 000000	LO.CSR 000006
AS.DIS= 000002	A.TRAN 000022	C.SSHT= 000004	FE.CEX= ***** GX	LO.CTL 000014
AS.DLT= 000001	CBLK1 000434R	C.SUB 000012	F\$SLVL= 000001	LO.LIN 000004
AS.EXT= 000004	CBLK3 000450R	C.SYST= 000006	F.NRBD= ***** GX	LO.PAR 000010
AS.FPA= 000001	CBLK4 000464R	C.SYTK= 000010	F.RSIZ= ***** GX	LO.PRI 000010
AS.PFA= 000004	CBLK5 000500R	C.TCB 000004	GS.DEL= 000001	LO.SCR 000014
AS.RCA= 000002	CEPTR 000746R	C.TIM 000006	G\$STPP= 000000	LO.STA 000015
AS.REA= 000005	CESYM 000466R	C.UIC 000016	G\$TSS= 000000	LO.VCT 000012
AS.RED= 000001	CEXNAM 000376R	DSIZE = ***** GX	G\$TTK= 000000	LR.AST 000026
AS.RRA= 000003	CEXPB 000406R	D\$AMXC 000072	G\$WRD= 000000	LR.CIR 000002
AS.WRT= 000002	CF.DDM= 000002	D\$AMXH 000074	G.CNT 000004	LR.CTL 000004
A\$CHK= 000000	CF.DYN= 000004	D\$ANN 000000	G.EFLG 000006	LR.EFN 000004
A\$CPS= 000000	CF.EIS= 000010	D\$BRPR 000102	G.GRP 000002	LR.LIN 000002
A\$PRI= 000000	CF.FRK= 100000	D\$BRTM 000100	G.LGTH= 000012	LR.PRM 000030
A\$TRP= 000000	CF.LOG= 000020	D\$DELF 000045	G.LNK 000000	LR.PRO 000002
A.ACC 000014	CF.LDM= 000001	D\$DELR 000046	G.STAT 000003	LR.STS 000000
A.ACCE 000000	CF.TIM= 000400	D\$END = 000104	IBLK1 000610R	002 LR.TCB 000002
A.AST 000006	CLKQB 000410R	D\$FNB 000034	IBLK2 000622R	002 LR.UNT 000005
A.BYT 000004	CL.OPN= ***** GX	D\$HIOR 000024	IP.ABO= 000040	LS.AST= 100000
A.CALL 000024	CM.CDS= 000004	D\$HOST 000022	IP.FAK= 000020	LS.CEX= 000001
A.CBL 000002	CM.CEN= 000003	D\$INAC 000044	IP.PND= 000100	LS.CIR= 000010
A.CONN 000012	CM.ELM= 000005	D\$INCT 000042	IP.JMR= 000200	LS.CXQ= 010000
A.DEQU 000002	CM.EXT= 000000	D\$JPL 000051	I\$RAR= 000000	LS.CX5= 000400
A.DIS 000002	CM.IND= 000000	D\$LI1 000020	I\$RDN= 000000	LS.DLM= 004000
A.DISC 000014	CM.INF= 000001	D\$LNAM 000006	I.AST 000022	LS.ECH= 001000
A.DQSR 177776	CM.LKT= 000007	D\$LNUM 000014	I.ATTL= 000044	LS.FDX= 004000
A.FLEN 000011	CM.MSG= 000011	D\$LST 000047	I.EFN 000003	LS.HDX= 010000
A.IBUF 000014	CM.RMT= 000010	D\$MAXC 000064	I.FCN 000012	LS.LIN= 000003
A.ILEN 000016	CP.DSB= 000010	D\$MAXH 000066	I.IOSB 000014	LS.LMC= 000007
A.IMAP 000012	CP.EXT= 000400	D\$MAXV 000070	I.LGTH= 000044	LS.NTI= 004000
A.INPU 000006	CP.LGO= 000004	D\$MLL 000040	I.LNK 000000	LS.ON = 000011
A.JOC 000003	CP.MSG= 000002	D\$MNJD 000041	I.LN2 000036	LS.OPT= 000400
A.JOS 000026	CP.NIO= 000100	D\$NA 000062	I.PRI 000002	LS.PRO= 000002
A.KSR5 177774	CP.NUL= 000001	D\$NBEA 000056	I.PRM 000024	LS.PWF= 040000
A.LGTH= 000014	CP.PRV= 000020	D\$NBRA 000054	I.TCB 000004	LS.TOP= 002000
A.LIN 000012	CP.RST= 000200	D\$NEND= 000054	I.UCB 000010	LS.UNF= 020000
A.MAS 000004	CP.SGL= 000040	D\$NLN 000030	I.XDBF 000040	LS.X25= 000012
A.MPCT 000011	C\$CKP= 000000	D\$NN 000060	I.XDBL 000044	LS.11D= 004000
A.NPR 000010	C\$ORE= 000400	D\$OUTT 000043	I.XIOP 000002	LX.CEX= 000004
A.NJM 000010	C\$RSH= 177564	D\$RETF 000050	I.XLEN= 000046	LX.LIN= 000006
A.OUTP 000010	C.AP5 000014	D\$RNN 000002	I.XLNK 000000	LX.PRO= 000005
A.PCB 000012	C.AST 000012	D\$RTMR 000076	I.XMOD 000006	L\$ASG= 000000

```

82                                     .SBTTL  CONSTANT DEFINITIONS
83
84                                     ;
85                                     ;  CONSTANTS
86                                     ;
87         000031         MAXCST = 25.          ; MAXIMUM LINE COST ALLOWED
88         000200         MAXNDC = 128.         ; MAXIMUM NUMBER OF NODE COUNTERS (LLC$DF)
89         000007         MAXWND = 7.           ; MAXIMUM WINDOW SIZE (SVC$DF)
90         000040         BLKSMN = 32.           ; MINIMUM BLOCK SIZE VALUE
91         002000         BLKSMX = 1024.         ; MAXIMUM BLOCK SIZE VALUE
92         000177         WNDSMX = 127.         ; MAXIMUM WINDOW SIZE VALUE
93         007777         CHNLMX = 4095.        ; MAXIMUM VALUE FOR CHANNEL NUMBER
94         000020         MAXID = 16.           ; Max length of net. mgmt. 'id-string'
95         000040         SPACE = 40.           ; ASCII space character
96
97                                     ;
98                                     ;  CHARACTERISTICS WORD 1 PROTOCOL BIT SELECTIONS
99                                     ;
100        000001         C1.DCP = 1             ; DDCMP
101        000002         C1.BSY = 2             ; BISYNC
102        000003         C1.SDL = 3             ; SDLC/ADCCP/HDLC
103        000004         C1.X25 = 4             ; X.25

```

65
 65
 65
 65

VCFG - VNP SCAN CONFIGURATION F MACRO V05.03b Monday 15-Jul-85 19:04^{K 4} Page 13-1
STORID - Store id-string according to system type

650 001430
651
652 001432 000261
653 001434

. RETURN
99\$: SEC
RETURN

; Signal string too long

;[TD001]
;[TD001]
;[TD001]
;[TD001]

VCFG - V
TPARS A

1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121

K 5

```

1065          .SBTTL TPARS ACTION ROUTINES
1066          :
1067          : TPARS ACTION ROUTINES
1068          :
1069          : DDM NAME (SLT$DF, DDM$DF, LLC$DF)
1070          :
1071 001512      L.NAME:
1072 001512      D.NAME:
1073 001512      S.NAME:
1074 001512 012700 000012'      MOV      #NAME,R0      ; POINT AT NAME WE'RE LOOKING FOR
1075 001512 016701 000000G      MOV      .PSTPT,R1      ; POINT AT NAME FROM CONFIG FILE
1076 001522 016702 000000G      MOV      .PSTCN,R2      ; GET CONFIG FILE NAME LENGTH
1077 001526 022702 000003      CMP      #3,R2      ; THREE CHARACTERS?
1078 001532 001407      BEQ      10$      ; IF EQ, YES
1079 001534 022702 000002      CMP      #2,R2      ; TWO CHARACTERS?
1080 001540 001011      BNE      20$      ; IF NE, NO - REJECT TRANSITION
1081 001542 122760 000040 000002      CMPB      #' ,2(R0)      ; IS OUR THIRD CHARACTER A SPACE?
1082 001550 001005      BNE      20$      ; IF NE, NO - REJECT TRANSITION
1083 001552 122021 10$      CMPB      (R0)+,(R1)+      ; SEE IF NAMES MATCH
1084 001554 001003      BNE      20$      ; FOR A LENGTH OF R2 CHARACTERS
1085 001556 005302      DEC      R2
1086 001560 001374      BNE      10$
1087 001562 000404      BR      30$      ; SUCCESS!
1088 001564      20$: REJ$      ; REJECT TRANSITION
1089 001574      30$: RETURN
1090          :
1091          : PROCESS LINE TABLE COUNT (LLC$DF, DLC$DF, DDM$DF)
1092          :
1093 001576      P.INC:
1094 001576 005767 000000G      TST      .MGMGE      ; MAPPED SYSTEM ?
1095 001602 100407      BMI      10$      ; IF MI, NO .. LEAVE
1096 001604 032767 000000G 000000G      BIT      #MS.INC,$MISS      ; IS THE LINE TABLE COUNT MISSING ?
1097 001612 001403      BEQ      10$      ; IF EQ, NO .. LEAVE
1098 001614 016767 000000G 000000G      MOV      .PNUMB,$$INC      ; GET THE LINE TABLE COUNT
1099 001522      10$: RETURN
1100          :
1101          : PROCESS CONTROLLER TABLE COUNT (DDM$DF, DLC$DF)
1102          :
1103 001624      D.EXT:
1104 001624 005767 000000G      TST      .MGMGE      ; MAPPED SYSTEM
1105 001630 100407      BMI      10$      ; IF MI, NO .. LEAVE
1106 001632 032767 000000G 000000G      BIT      #MS.EXT,$MISS      ; IS THE CONTROLLER TABLE COUNT MISSING ?
1107 001640 001403      BEQ      10$      ; IF EQ, NO .. LEAVE
1108 001642 016767 000000G 000000G      MOV      .PNUMB,$$EXT      ; ELSE, GET THE CONTROLLER TABLE COUNT
1109 001650      10$: RETURN
1110          :
1111          : LLC PROCESS EXTENSION
1112          :
1113 001652      P.EXT:
1114 001652 005767 000000G      TST      .MGMGE      ; MAPPED SYSTEM ?
1115 001656 100407      BMI      10$      ; IF MI, NO .. LEAVE
1116 001660 016767 000000G 000000G      MOV      .PNUMB,$$EXT      ; GET THE LLC PROCESS EXTENSION
1117 001666 026727 000000G 000200      CMP      .PNUMB,#128.      ; IS THE PROCESS EXTENSION >= 4K ?
1118 001674 103001      BHIS      101$      ; IF HIS, YES .. ERROR
1119 001676      10$: RETURN

```

L 5

```

1598                                     .SBTTL LNKEND - LINK BLOCK AT END OF LIST
1599
1600                                     ;+
1601                                     : LNKEND - LINK BLOCK AT END OF LIST
1602                                     :
1603                                     : INPUTS:
1604                                     : R0 - ADDRESS OF LISTHEAD
1605                                     : R5 - UNMAPPED ADDRESS OF CURRENT BLOCK
1606                                     :
1607                                     : OUTPUTS:
1608                                     : BLOCK IS LINKED AT END OF LIST
1609                                     :
1610                                     : -
1611 LNKEND::GETRV R0,#2 ; READ IN LISTHEAD
1612 004030 MOV R0,R1 ; SAVE ADDRESS OF PREVIOUS BLOCK
1613 004046 010001 000000G 10$: MOV $BFR,R0 ; GET ADDRESS OF NEXT BLOCK IN LIST
1614 004050 016700 BEQ 20$ ; BR IF END OF LIST - LINK IT HERE
1615 004054 001414 CEACC$ R0 ; CONVERT TO MAPPED ADDRESS
1616 004056 GETAD R0,#2 ; READ IN LISTHEAD
1617 004066 BR 10$ ; GET NEXT BLOCK IN LIST
1618 004104 000760 20$: MOV R5,-(SP) ; GET UNMAPPED ADDR OF CURRENT BLOCK
1619 004106 010546 MOV R1,-(SP) ; GET ADDRESS OF PREVIOUS BLOCK
1620 004110 010146 CALL $XLINK ; LINK BLOCK INTO LIST
1621 004112
1622 004116 RETURN

```

VCFG
MACRO
MACRO

TRANS

VNPBP1
XPDD8

VCFG CREATED BY MACRO ON 15-JUL-85 AT 19:06 PAGE 8 K 7

MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES									
	#24-1035	#24-1037	#24-1040	#24-1043	#24-1045	#24-1048	#24-1051	#24-1054	#24-1057	#24-1059
	#24-1061									
TRANS	#5-59	#17-724	#17-726	#17-728	#17-730	#17-732	#17-734	#17-736	#17-738	#17-740
	#17-742	#17-744	#17-746	#17-747	#17-749	#17-750	#17-752	#17-753	#17-755	#17-756
	#17-758	#17-759	#17-761	#17-762	#17-763	#17-765	#17-766	#17-768	#17-770	#17-772
	#17-779	#17-780	#17-781	#17-782	#17-784	#17-785	#17-786	#17-788	#17-789	#17-790
	#18-797	#18-800	#18-803	#18-806	#18-809	#18-810	#18-813	#18-814	#18-817	#18-820
	#18-821	#18-824	#18-827	#18-828	#18-831	#19-840	#19-842	#19-848	#19-850	#20-858
	#20-859	#20-862	#20-865	#20-868	#20-871	#20-874	#20-875	#20-878	#20-881	#20-882
	#20-885	#21-892	#21-894	#21-896	#21-898	#21-900	#21-902	#21-904	#21-908	#22-933
	#22-935	#22-937	#22-939	#22-941	#22-943	#22-945	#22-946	#22-948	#22-949	#22-952
	#22-953	#22-956	#22-957	#22-960	#22-961	#22-964	#22-965	#22-968	#22-969	#22-972
	#22-975	#22-981	#22-982	#23-989	#23-992	#23-995	#23-998	#23-1001	#23-1004	#23-1007
	#23-1010	#23-1013	#24-1020	#24-1022	#24-1024	#24-1026	#24-1028	#24-1030	#24-1032	#24-1034
	#24-1036	#24-1038	#24-1039	#24-1041	#24-1042	#24-1044	#24-1046	#24-1047	#24-1049	#24-1050
	#24-1052	#24-1053	#24-1055	#24-1056	#24-1058	#24-1060				
VNPBPT	#8-259	8-260								
XPDDBS	#5-60	5-63								

```

409          .SBTTL OPEN CONFIG FILE
410          ;+
411          ; OPEN - OPEN THE CONFIG FILE
412          ;
413          ; INPUTS:
414          ;     NONE
415          ;
416          ; OUTPUTS:
417          ;     C-BIT=SUCCESS/FAILURE
418          ; -
419 000400 012700 000000G OPEN: MOV    #$DFUIC,R0    ; SETUP ASCII FILE SPEC
420 000404 012701 000000'    MOV    #CFGNAM,R1    ; FOR CETAB.MAC
421 000410 012702 000006'    MOV    #CFGEXT,R2    ;
422 000414          CALL    $AZFS          ; ...
423 000420          CALL    $CLOUE          ; OPEN THE CONFIG FILE
424 000424 105777 000000G    TSTB    @CLIOS      ; SUCCESS? (CLEARS C-BIT)
425 000430 100401          BMI     101$      ; IF MI, NO
426 000432          RETURN
427
428          ;
429          ; ERROR CONDITION
430          ;
431 000434 101$:  MSG$R  1P          ; OPEN FAILURE
  
```

```

783                                     .SBTTL  SPFILL - Fill area with spaces
784                                     ;+
785                                     *** SPFILL - Fill area with spaces
786                                     ;
787                                     INPUTS
788                                     R0 - address of area to fill
789                                     R1 - length of area to fill
790                                     ;
791                                     OUTPUTS
792                                     All registers preserved
793                                     ;
794                                     SPFILL::
795                                     MOV     R0,-(SP)           ; save registers
796                                     MOV     R1,-(SP)
797                                     BEQ     20$               ; nothing to do?
798                                     10$:   MOVB   #SPACE,(R0)+
799                                     DEC     R1
800                                     BGT     10$
801                                     20$:   MOV     (SP)+,R1       ; restore regs
802                                     MOV     (SP)+,R0
803                                     RETURN
804                                     .END
805                                     ***-1

```

```

129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146 000000
147 000000 005067 000000'
148 000004
149 000010 105777 000000G
150 000014 100420
151 000016 052010
152 000020 001422
153 000022
154 000026 103402
155 000030
156 000034
157 000040 005767 000000'
158 000044 001401
159 000046 000261
160 000050 016700 000000'
161 000054
162
163
164
165
166 000056 012767 000002 000000'
167 000064 000770
168 000066 012767 000006 000000'
169 000074 000757

;+
; $CTBIO - CETAB I/O
; THIS ROUTINE IS CALLED TO TRANSFER THE CETAB TASK IMAGE INTO CORE.
; INPUTS:
;   R1=FILE SPEC ADDRESS
;   R2=FILE SPEC LENGTH
; OUTPUTS:
;   C-BIT=SUCCESS/FAILURE
;   C=1
;       R0=ERROR CODE
;   C=0
;       R1 = CETAB ALLOCATION ADDRESS
;       R2 = CETAB ALLOCATION LENGTH
;
; $CTBIO::
;   CLR    ERROR           ; NO ERROR, EITHER
;   CALL   $CLOPE          ; OPEN PROCESS FILE
;   TSTB   @%CLIOS         ; LOOK AT GCL IOSB
;   BMI    101$            ; IF M1, FAILURE
;   BIS    (R0)+.(R0)      ; IS FILE CONTIGUOUS?
;   BEQ    111$            ; IF EQ, NO
;   CALL   PR.LBL          ; READ LABEL BLOCK
;   BCS    10$             ; IF CS, ERROR
;   CALL   PR.XFR          ; READ PROCESS TASK IMAGE INTO CORE
;   CALL   $CLDEQ          ; CLOSE PROCESS FILE
;   TST    ERROR           ; ANY ERRORS? (CLEARS C-BIT)
;   BEQ    30$             ; IF ZERO, NO
;   SEC    30$             ; INDICATE FAILURE
;   MOV    ERROR,R0        ; GET ERROR CODE
;   RETURN
;
; ; ERROR CONDITIONS
;
; 101$: MOV    #ERR2,ERROR  ; OPEN FAILURE
;       BR     20$
; 111$: MOV    #ERR6,ERROR  ; NOT CONTIGUOUS
;       BR     10$

```

VCLOP
CLORE

.SBTIL CLOREM - REMOVE ENTRIES FROM CLOCK QUEUE

*** - CLOREM - REMOVE ENTRIES FROM CLOCK QUEUE

INPUT:

R5 = TCB ADDRESS OF ENTRIES TO REMOVE FROM CLOCK QUEUE IF SET
 EXECUTOR STATE OFF COMMAND

OUTPUT:

IF SET EXECUTOR STATE OFF, THE NTINIT ENTRY IS REMOVED FROM THE
 CLOCK QUEUE IF IT IS IN THE LIST. IF CLEAR SYSTEM, THE 1 SECOND TIMER
 AND 50 MSEC TIMER ENTRIES ON THE CLOCK QUEUE ARE REMOVED FROM THE LIST
 IF ANY EXIST.

CLOREM::

CLR AUXPDV ; INITIALIZE AUX PDV ADDRESS
 MOV #*RAUX,R0 ; GET AUX PROCESS NAME
 CALL FNDPDV ; GET PDV ADDRESS
 BCS 5\$; BR IF NOT FOUND
 MOV R0,AUXPDV ; SAVE PDV ADDRESS
 5\$: MOV #SBFR,R4 ; POINT AT I/O BUFFER
 MOV .CLKHD,R0 ; POINT AT CLOCK QUEUE LISTHEAD
 GETRV R0,#C.LGTH ; READ IN LISTHEAD

; SCAN ENTIRE CLOCK QUEUE

10\$: MOV R0,R3 ; SAVE ADDRESS OF PREVIOUS ENTRY
 MOV (R4),R0 ; GET ADDRESS OF NEXT ENTRY
 BEQ 20\$; IF EQ, END OF LIST
 GETRV R0,#C.LGTH ; READ IN ENTRY
 TST R5 ; CHECK FOR NTINIT ENTRY?
 BEQ 12\$; BR IF NO
 CMP R5,C.TCB(R4) ; IS THIS ENTRY TO BE REMOVED ?
 BEQ 15\$; IF EQ, YES
 CMP #LS.NOD,RQB ; SET EXECUTOR COMMAND?
 BEQ 10\$; BR IF YES (ELSE MUST BE DLX ONLY SYSTEM)
 12\$: CMPB #C.SYST,C.RQT(R4) ; SINGLE SHOT INTERN SYSTEM SUBROUTINE?
 BNE 10\$; BR IF NO
 MOV .PDVTA,R1 ; GET STARTING ADDRESS OF CEXCM
 CMP R1,R0 ; IS THIS ENTRY FROM CEXCM?
 BHI 10\$; BR IF NO
 ADD #<\$CEXND-\$PDVTA>,R1 ; POINT TO END OF CEXCM
 CMP R1,R0 ; IS ENTRY FROM CEXCM?
 BLO 10\$; BR IF NO
 TST .MGMGE ; MAPPED SYSTEM?
 BMI 15\$; BR IF NO - REMOVE ENTRY
 MOV AUXPDV,R1 ; GET ADDRESS OF AUX PDV
 BEQ 10\$; BR IF AUX NOT IN SYSTEM
 CMP C.AR5(R4),Z.DSP(R1) ; IS THIS ENTRY MAPPED TO AUX?
 BNE 10\$; BR IF NO

; ENTRY TO BE REMOVED

15\$: MOV C.TCB(R4),-(SP) ; COPY POSSIBLE NTINIT TCB ADDRESS

69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86 000002
 87 000002 005067 177772
 88 000006 012700 004640
 89 000012
 90 000016 103402
 91 000020 010067 177754
 92 000024 012704 000000G
 93 000030 016700 000000G
 94 000034
 95
 96
 97
 98 000052 010003
 99 000054 011400
 100 000056 001470
 101 000060
 102 000076 005705
 103 000100 001407
 104 000102 020564 000004
 105 000106 001432
 106 000110 022767 000014 000000G
 107 000116 001755
 108 000120 122764 000006 000002 12\$:
 109 000126 001351
 110 000130 016701 000000G
 111 000134 020100
 112 000136 101345
 113 000140 062701 000000C
 114 000144 020100
 115 000146 103741
 116 000150 005767 000000G
 117 000154 100407
 118 000156 016701 177616
 119 000162 001733
 120 000164 026461 000014 000000G
 121 000172 001327
 122
 123
 124
 125 000174 016446 000004

```
54  
55  
56  
57  
58  
59  
60  
61  
62 000000
```

```
      .SBTTL  MACRO DEFINITIONS  
      ;**  
      ; LIBRARY MACROS  
      ;**  
      .MCALL  EMSG$,EMSG$,NTLER$,GETRV,PUTRC,GETAD,PUTAD,UCBDF$  
      .MCALL  SWTCHO,SWTCHI  
      .ENABL  LC  
      UCBDF$      ; DEFINE UCB OFFSETS
```



```

579 001546 012667 000176'      MOV      (SP)+,PLEN      ; AND RESTORE LENGTH
580 001552 012767 000002G 000000G  MOV      #D.PCB+2,$RSIZE ; SET LENGTH OF TRANSFERS
581 001560 012702 000204'      MOV      #DEVBUF,R2      ; SET BASE REGISTER
582 001564 005067 000000G      CLR      $RADDR      ; SET NO PREVIOUS DCB
583 001570 016767 000000G 000204'  MOV      $DEVHDL,DEVBUF ; STORE FIRST DCB ADDRESS
584 001576 016701 000000G      MOV      $RADDR,R1      ; STORE PREVIOUS DCB ADDRESS
585 001602 016767 000204' 000000G  MOV      DEVBUF,$RADDR ; LOAD NEXT DCB ADDRESS
586 001610 001403      BEQ      40$      ; END OF LIST
587 001612      GETRV      ; READ IN DCB IMAGE
588 001616 000767      BR      30$      ; CONTINUE
589
590 001620 016746 000000G      40$:  MOV      $RADDR,-(SP) ; SAVE CURRENT DCB ADDRESS
591 001624 005701      TST      R1      ; IS THERE A PREVIOUS DCB ?
592 001626 001004      BNE      44$      ; YES
593 001630 016767 000174' 000000G  MOV      PBUF,$DEVHDL ; SET THIS DCB AS FIRST IN LIST
594 001636 000411      BR      46$      ; AND CONTINUE
595
596 001640      44$:  GETRV      R1      ; READ IN PREVIOUS DCB
597 001650 016767 000174' 000204'  MOV      PBUF,DEVBUF ; AND POINT IT AT OURS
598 001656      PUTRC      ; REWRITE DCB TO SYS IMG
599
600 001662      46$:  GETRV      PBUF      ; REREAD OUR DCB
601 001674 012667 000204'      MOV      (SP)+,DEVBUF ; POINT AT NEXT DCB
602 001700      PUTRC      ; REWRITE OUR DCB
603 001704      50$:  RETURN      ; BACK TO USER MODE
604
605      ;
606      ; ERROR CONDITIONS
607      ;
608 001706 116777 000000G 000000G 101$: MOV      $IOSB,$CLIOS ; COPY ERROR CODE
609 001714      MSG$R      2M      ; READ FAILURE
610 001722      111$:  MSG$R      2R      ; D.UNIT > D.UNIT+1
611 001730      121$:  MSG$R      2Q      ; D.UCB FIELD OUTSIDE TABLE LIMITS
612 001736 006202      131$:  ASR      R2      ; ODD ADDRESS?
613 001740 103003      BCC      132$      ; IF CC, NO
614 001742      MSG$R      2S      ; UCB ADDRESS ODD
615 001750      132$:  MSG$R      2T      ; UCB ADDRESS OUTSIDE TABLE LIMITS
616 001756      141$:  MSG$R      2U      ; U.SCB FIELD OUTSIDE TABLE LIMITS
617 001764 006202      151$:  ASR      R2      ; ODD ADDRESS?
618 001766 103003      BCC      152$      ; IF CC, NO
619 001770      MSG$R      2V      ; SCB ADDRESS ODD
620 001776      152$:  MSG$R      2W      ; SCB ADDRESS OUTSIDE TABLE LIMITS
621
622      ;
623      ; RELOCATE AND VALIDATE DATA STRUCTURE ADDRESS
624      ;
625 002004 166702 000170'      RVAD:  SUB      BASE,R2      ; ADJUST BY TKB BASE ADDRESS
626 002010 066702 000000G      ADD      $LBUF,R2      ; AND BY BUFFER ADDRESS FROM THE POOL
627 002014 066702 000200'      VAD:  ADD      VREL,R2      ; RELOCATE TO TRUE ADDRESS IN SCOM
628 002020 032702 000001      BIT      #1,R2      ; IS ADDRESS ODD?
629 002024 000261      SEC      ; IN CASE IT IS
630 002026 001005      BNE      10$      ; IF NE, YES
631 002030 020267 000174'      CMP      R2,PBUF      ; IS ADDRESS < START OF BUFFER?
632 002034 103402      BCS      10$      ; IF CS, YES
633 002036 026702 000202'      CMP      TABEND,R2 ; SET C-BIT IF ADDRESS > END OF BUFFER
634 002042 006146      10$:  ROL      -(SP)      ; SAVE C-BIT
635 002044 166702 000200'      SUB      VREL,R2      ; BIAS ADDRESS BY DIFFERENTIAL

```

VDIS - VNP SHOW COMMANDS
Table of contents

MACRO V05.03b Monday 15-Jul-85 12:14 K 14

5-	54	MACRO CALLS	
6-	105	LOCAL DATA	
7-	278	LOCAL SYMBOLS	
8-	327	\$DSYS - SHO SYS	
9-	367	\$DOBJ - SHO OBJ	
10-	439	\$DALI - SHOW ALIAS	
11-	555	\$DPRO - SHOW PROCESS	
12-	645	\$DLOG - SHOW LOGGING	
13-	799	DLOTYP - SHOW LOGGING SINK TYPE	
14-	843	DLOSTA - DISPLAY LOGGING STATE	
15-	867	DLONAM - DISPLAY LOGGING SINK NAME	
16-	933	CVTDEV - CONVERT DEVICE NAME TO ASCII	
17-	958	CHKSNK - CHECK FOR MATCH ON SINK TYPE	
18-	1002	DLOSINK - DISPLAY LOGGING SINK NODE	
19-	1075	DLOEVT - DISPLAY EVENT CHARACTERISTICS	
20-	1173	BLDMSK - BUILD EVENT MASK	
21-	1203	BLDLIN - BUILD LINE ENTITY	
22-	1272	BLDNOD - BUILD NODE ENTITY	
23-	1329	CKPREV	
24-	1353	CHKBIT - CHECK TO SEE IF BIT IS ON	
25-	1381	FNDADD - FIND REMOTE NODE BY ADDRESS	
26-	1416	\$DNOD - SHOW NODE	
27-	1907	\$DCIR - SHOW CIRCUIT	
28-	2139	\$DLIN - SHOW LINE	
29-	2243	\$DMOD - SHOW MODULE INFORMATION	
30-	2292	DMXAC - SHOW MODULE X25-ACCESS	
31-	2316	DMXACD - Display X25-ACCESS destinations	;[TD001]
32-	2387	NXTRDT - Get next RDT block	;[TD001]
33-	2413	DMXACN - Display X25-ACCESS networks	;[TD001]
34-	2513	DMXPR - SHOW MODULE X25-PROTOCOL	
35-	2619	DMDTE - SHOW MODULE X25-PROTOCOL DTES	
36-	2687	NXTDTE - GET NEXT DTE BLOCK	
37-	2725	DMGRP - SHOW MODULE X25-PROTOCOL GROUPS	
38-	2796	NXTGRP - Get next group block	;[TD001]
39-	2836	DMX5S - SHOW MODULE X25-SERVER	
40-	2909	DMDST - SHOW MODULE X2n-SERVER DESTINATION	;[TD001]
41-	3112	NXTDST - Get next destination block	;[TD001]
42-	3153	DMX9S - SHOW MODULE X29-SERVER	
43-	3197	CKTYPE - CHECK TYPE	
44-	3236	FNDBLK - Find next block meeting given criteria	;[TD001]
45-	3337	NXTBLK - READ IN NEXT BLOCK	
46-	3371	NXTCIR - GET NEXT CIRCUIT	
47-	3403	NXTLIN - GET NEXT LINE	
48-	3432	DISOBJ - DISPLAY OBJECT INFO	
49-	3533	DISALI - DISPLAY ALIAS INFO	
50-	3671	DISPRO - DISPLAY PROCESS INFORMATION	
51-	3734	DISNOD - DISPLAY NODE INFORMATION	
52-	3857	DISLOO - DISPLAY LOOPBACK NODE INFORMATION	
53-	3924	DISCIR - DISPLAY CIRCUIT INFORMATION	
54-	4316	DISPVC - DISPLAY PVC CIRCUIT INFORMATION	;[TD001]
55-	4375	DISLIN - DISPLAY LINE INFORMATION	
56-	4552	FMTLIN - FORMAT LINE-ID	
57-	4603	FMTCIR - FORMAT CIRCUIT-ID	
58-	4667	GTEXEC - GET EXECUTOR ADDRESS AND NAME	
59-	4698	GTSTID - GET NODE STATE AND IDENTIFICATION	
60-	4761	TSKSKCH - SEARCH TCB'S FOR SPECIFIED TASK	
61-	4792	FNDPDV - FIND PDV ADDRESS	

VDIS - VNP SHOW COMMANDS
\$DIALI - SHOW ALIAS

MACRO V05.03b Monday 15-Jul-85 12:14 Page 10-1

K 15

```

496 002702 062700 000002      ADD    #LR.ALI,R0      ; POINT TO ALIAS NAME
497 002706                  CALL  NXTBLK      ; GET NEXT ALIAS NAME BLOCK IN LIST
498 002712 103424              BCS    90$      ; BR IF END OF LIST
499 002714 010302              MOV    R3,R2      ; POINT TO ALIAS NAME BLOCK
500 002716 062702 000002      ADD    #A.NAM,R2    ; POINT TO ALIAS NAME
501 002722 012701 000006      MOV    #6,R1      ; 6 CHARACTERS MAXIMUM
502 002726 122022              70$: CMPB   (R0)+,(R2)+ ; IS THIS THE ONE WE ARE SEARCHING FOR
503 002730 001362              BNE     60$      ; BR IF NO
504 002732 005301              DEC     R1      ; CHECKED ALL CHARACTERS?
505 002734 003374              BGT     70$      ; BR IF NO
506 002736 005767 000000G      TST     $OPTON    ; WAS A SCOPE SPECIFIED
507 002742 001404              BEQ     80$      ; BR IF NO
508 002744 026063 000006 000010 CMP    LR,UCB-LR.DES(R0),A.UCB(R3) ; DO THE SCOPES MATCH?
509 002752 001351              BNE     60$      ; BR IF NO
510 002754              80$: CALL    DJSALI      ; DISPLAY THE ALIAS INFORMATION
511 002760 005204              INC     R4      ; DON'T DISPLAY HEADER
512 002762 000745              BR      60$      ; SEE IF ANY MORE
513 002764 005704              90$: TST     R4      ; WAS ONE FOUND?
514 002766 001426              BEQ     130$     ; BR IF NO
515 002770 000443              BR      150$     ; FINISHED
516                  ;
517                  ; SHOW KNOWN ALIASES
518
519 002772 005767 000000G      100$: TST     $OPTON    ; WAS SCOPE SPECIFIED?
520 002776 001431              BEQ     140$     ; BR IF NO
521 003000 012700 000151'      MOV    #AKNOWN,R0 ; HEADER FOR KNOWN ALIASES
522 003004 112021              110$: MOVB   (R0)+,(R1)+ ; STORE IT
523 003006 001376              BNE     110$     ;
524 003010 005004              CLR     R4      ; INDICATE HEADER TO BE PRINTED
525 003012 012700 000000G      120$: MOV    #RQB,R0 ; GET INPUT PARAMETERS
526 003016              CALL    NXTBLK    ; GET NEXT ALIAS NAME BLOCK
527 003022 103426              BCS    150$     ; BR IF AT END OF LIST
528 003024 026063 000016 000010 CMP    LR,UCB(R0),A.UCB(R3) ; DO THE SCOPES MATCH?
529 003032 001367              BNE     120$     ; BR IF NO
530 003034              CALL    DJSALI      ; DISPLAY INFO FOR THE ALIAS NAME
531 003040 005204              INC     R4      ; INDICATE HEADER NOT TO BE DISPLAYED
532 003042 000763              BR      120$     ; CONTINUE
533
534 003044              130$: DIR$    #TIDET      ; DETACH TERMINAL
535 003052              ERRPT$  AERR1,$EROUT    ; ALIAS NOT IN SYSTEM
536
537 003062              140$: DIR$    #TIDET      ; DETACH TERMINAL
538 003070              ERRPT$  AERR2,$EROUT    ; SCOPE NOT SPECIFIED
539
540 003100              150$: RETURN      ; FINISHED
541
542                  .ENABL  LC
543
544                  ;
545                  ; ERROR MESSAGES
546                  ;
547 003102              ERMSG$    <Alias not in system>
548 003125              ERBLK$    AERR1
549
550 003130              ERMSG$    <Parameter missing, Scope>
551 003160              ERBLK$    AERR2
552

```

```

1002 .SBTTL DLOSNK - DISPLAY LOGGING SINK NODE
1003
1004 +
1005 DLOSNK - DISPLAY LOGGING SINK NODE
1006
1007 INPUTS:
1008 TEMP3 - SINK NODE ADDRESS TO DISPLAY
1009
1010 OUTPUTS:
1011 LOGGING SINK NODE DISPLAYED ON USER'S TERMINAL
1012 R0,R1,R2 DESTROYED
1013
1014 -
1015 005360 004567 000000G DLOSNK: JSR R5,$SAVRG ; SAVE R3,R5
1016 005364 012702 001062 MOV #TEMP2,R2 ; GET ADDRESS FOR NODE NAME
1017 005370 016701 173606 MOV TEMP3,R1 ; GET SINK NODE ADDRESS
1018 005374 001403 BEQ 10$ ; BR IF EXECUTOR
1019 005376 020127 177777 CMP R1,#-1 ; IS SINK NODE $HOST ?
1020 005402 001034 BNE 20$ ; IF NOT - BRANCH
1021
1022 005404 10$: SWITCH #HMBUF ; SET INPUT BUFFER
1023 005412 GETRV $DECTP,#D$END ; READ DECNET HOME BLOCK
1024 005432 SWITCH
1025
1026 005440 016701 000022G MOV $HMBUF+D$HOST,R1 ; GET HOST NODE ADDRESS
1027 005444 020167 000014G CMP R1,$HMBUF+D$LNUM ; IS HOST SAME AS EXECUTOR
1028 005450 001011 BNE 20$ ; IF NOT - BRANCH
1029
1030 005452 016701 000014G MOV $HMBUF+D$LNUM,R1 ; GET EXECUTOR NODE ADDRESS
1031 005456 016722 000006G MOV $HMBUF+D$LNAM+0,(R2)+ ; STORE EXECUTOR NODE NAME
1032 005462 016722 000010G MOV $HMBUF+D$LNAM+2,(R2)+ ; ...
1033 005466 016722 000012G MOV $HMBUF+D$LNAM+4,(R2)+ ; ...
1034 005472 000413 BR 30$ ; CONTINUE
1035 005474 20$: CALL FNDADD ; FIND REMOTE NODE NAME BLOCK
1036 005500 103416 BCS 40$ ; BR IF NOT FOUND
1037 005502 012700 001034 MOV #TEMP1,R0 ; POINT TO NODE NAME BLOCK
1038 005506 016022 000002 MOV R.NAM+0(R0),(R2)+ ; STORE NODE NAME
1039 005512 016022 000004 MOV R.NAM+2(R0),(R2)+ ; ...
1040 005516 016022 000006 MOV R.NAM+4(R0),(R2)+ ; ...
1041 005522 124227 000040 30$: CMPB -(R2),#40 ; DELETE TRAILING BLANKS
1042 005526 001775 BEQ 30$ ; ...
1043 005530 005202 INC R2 ; POINT PAST LAST NON-BLANK
1044 005532 105022 CLRB (R2)+ ; CREATE ASCIZ STRING
1045 005534 000402 BR 50$ ; CONTINUE
1046 005536 005067 173320 40$: CLR TEMP2 ; NO NODE NAME
1047
1048 005542 010167 173266 50$: MOV R1,TEMP1 ; POINT TO STORAGE FOR NODE ADDRESS
1049 005546 042701 000000G BIC #^CARAMSK,R1 ; GET AREA NUMBER
1050 005552 042767 000000G BIC #ARAMSK,TEMP1 ; STRIP OF AREA BITS
1051
1052 005560 026727 173416 177777 CMP TEMP3,#-1 ; $HOST SINK NODE ?
1053 005566 001022 BNE 70$ ; IF NOT - BRANCH
1054
1055 005570 000241 CLC ; DO NOT ROR C-BIT
1056 000012 .REPT 10.
1057 ROR R1 ; CALCULATE AREA NUMBER
1058 .ENDM

```

```
396 .SBTTL CEXYM - LOOK FOR CEX SYMBOLS
397
398 ;+
399 ; CEXYM - LOOK FOR SPECIAL COMM EXEC SYMBOLS
400
401 ; INPUTS:
402 ;     NONE
403
404 ; OUTPUTS:
405 ;     C-BIT=SUCCESS/FAILURE
406 ;     R0,R1,R2=DESTROYED
407 ; -
408
409 000466 112767 000600G 000000G CEXYM: CALL $CLSA, ; SAVE GCL CONTEXT
410 000472 112767 000600G 000000G MOVB #CL.OPN,$CLFLG ; LEAVE FILE OPEN
411
412 000500 012700 000000G MOV #SDFUIC,R0 ; SETUP ASCIZ FILE SPEC
413 .IF NDF R$SMPL ;
414 000504 012701 000376' MOV #CEXNAM,R1 ; FOR [XXX,54]CEX.STB
415
416 .IFF
417 MOV #RSX,R1 ; FOR [XXX,54]RSX11M.STB
418
419 .IFTF
420 000510 012702 000402' MOV #STBEXT,R2 ;
421 000514 000514 CALL $AZFS ; ...
422
423 .IFT
424 000520 012700 000434' MOV #CBLK1,R0 ; GET LIST OF SYMBOLS
425
426 .IFF
427 MOV #XBLK1,R0 ; GET FIRST LOOKUP CONTROL BLOCK ADDRESS
428
429 .IFTF
430 000524 000524 CALL $SYMBL ; LOOK FOR SYMBOLS IN STB FILE
431 000530 103453 BCS STBERR ; IF CS, FILE ERROR
432
433 .ENDC
434
435 000532 012700 000000G MOV #SDFUIC,R0 ; SETUP ASCIZ FILE SPEC
436
437 .IF DF R$SMPL ;
438
439 MOV #RSX,R1 ; FOR [XXX,5X]RSX11M.STB
440
441 000536 012701 000376' MOV #CEXNAM,R1 ; FOR [XXX,54]CEX.STB
442
443 .ENDC
444
445 000542 012702 000402' MOV #STBEXT,R2 ;
446 000546 000546 CALL $AZFS ; ...
447 000552 012700 000610' MOV #IBLK1,R0 ; GET FIRST LOOKUP CONTROL BLOCK ADDRESS
448 000556 000556 CALL $SYMBL ; LOOK FOR SYMBOLS IN STB FILE
449 000562 103436 BCS STBERR ; IF CS, FILE ERROR
450 000564 012700 000000G MOV #SDFUIC,R0 ; SETUP ASCIZ FILE SPEC
451 000570 012701 000000G MOV #SCET,R1 ; FOR [XXX,5X]CETAB.STB
452 000574 012702 000402' MOV #STBEXT,R2 ; ...
```

```

L$SDRV= 000000      P$SP45= 000000      T.LDV 000044      UA.TRN= 001000      $PDVNM= ***** GX
L$SP11= 000001      P$SWRD= 000000      T.LNK 000000      UA.TYP= 000010      $PDVTA= ***** GX
L$S11R= 000000      P.BLKS 000016      T.MXSZ 000050      VF.SS = ***** GX      $SPRCLN 000374RG 002
MSGB 000254R      002 P.BUSY 000024      T.NAM 000006      VXBLK1 000552R      $SPREXT 000374RG 002
MSGOUT 002070R      P.TOC 000003      T.OFF 000066      VXBLK2 000564R      $SPRINC 000374RG 002
M$SCRB= 000124      P.LNK 000000      T.PCB 000046      VXBLK3 000576R      002 $PRIO = ***** GX
M$SCRX= 000000      P.MAIN 000012      T.PRI 000002      V$CTR= 001000      $SPLEN 000374RG 002
M$SFCS= 000000      P.NAM 000004      T.RVFL 000012      XBLK1 000514R      $SPRLN 000374RG 002
M$SMGE= 000000      P.OWN 000026      T.RRFL 000072      XBLK2 000526R      002 $PUTRC= ***** GX
M$SNET= 000000      P.PRI 000002      T.SAST 000054      XBLK3 000540R      002 $RADDR= ***** GX
M$SOVR= 000000      P.REL 000014      T.SRCT 000071      X$SDBT= 000000      $SAVAL= ***** GX
NLBLK 000666R      P.SIZE 000016      T.STAT 000032      $ALL15= ***** GX      $SCEX 000000RG
NTINAM 000430R      002 P.STAT 000030      T.ST2 000034      $A7FS = ***** GX      $SLTMA= ***** GX
N$SACC= 000001      P.SUB 000010      T.ST3 000036      $BFR = ***** GX      $SLTNM= ***** GX
N$SBUF= 000001      P.SWSZ 000022      T.TCBL 000030      $CCBNM= ***** GX      $SYMBL= ***** GX
N$SLDV= 000001      P.TCB 000026      T.T10 000057      $CEAVL= ***** GX      $TIOUT= ***** GX
N$SMCP= 000001      P.WAIT 000020      T.TKSZ 000060      $CET = ***** GX      $VFLAG= ***** GX
N$SMML= 000001      Q$SOP= 000010      T.UCB 000026      $CLFLG= ***** GX      $VIMI = ***** GX
N$SMOV= 000010      RUN 001206R      T2.ABO= 000100      $CLSAV= ***** GX      $$ = 177777
N$SNC= 000001      RUNNTI 001504R      T2.AST= 100000      $CTBIO= ***** GX      $SMUX = ***** GX
N$SPEM= 000001      R$SDER= 000000      T2.CAF= 000400      $CXOPT= ***** GX      $$$ = 000226R 002
O.AST 000006      R$SK11= 000001      T2.CHK= 020000      $DEA16= ***** GX      $$$ = 000252R 002
O.EFN 000010      R$SND= 000000      T2.CKD= 010000      $DECHM= ***** GX      .ABTIM= ***** GX
O.ESB 000012      R$S11M= 000000      T2.DST= 040000      $DECPT= ***** GX      .CEAVL= ***** GX
O.LGTH= 000034      SLTTO 000672R      T2.FXD= 002000      $DFUIC= ***** GX      .CEPWR= ***** GX
O.LNK 000000      STBERR 000660R      T2.HLT= 000200      $ERMSG= ***** GX      .CLINS= ***** GX
O.MCRL 000002      STBEXT 000402R      T2.REX= 001000      $ERR44= ***** CX      .CLKHD= ***** GX
O.PTCB 000004      SYSFDB= ***** GX      T2.SEF= 004000      $ERR45= ***** GX      .CXALL 000062
O.STAT 000014      S$SWRG= 000000      T2.SPN= 000004      $ERR46= ***** GX      .CXCSR 000004
PC.AL= 000004      S$YSZ= 007600      T2.STP= 000020      $ERR47= ***** GX      .CXLBR 000060
PC.ALH= 000001      TICKS = 000001      T2.WFR= 000001      $ERR48= ***** GX      .CXLB1 000030
PC.HIH= 000001      TSKERR 000452R      T3.ACP= 100000      $ERR51= ***** GX      .CXLB2 000044
PC.LOW= 000002      TSKSCH 001120R      T3.CAL= 000100      $ERR53= ***** GX      .CXLEN 000064
PC.NRM= 000400      TS.BLK= 171700      T3.CLI= 001000      $ERR54= ***** GX      .CXPCB 000006
PC.XAF= 000010      TS.CKP= 000200      T3.GFL= 000010      $ERR55= ***** GX      .CXSYM 000010
PC.XIT= 000200      TS.CKR= 000100      T3.MCR= 004000      $ERR56= ***** GX      .CXUNL 000024
PDVTOT 000704R      002 TS.EXE= 100000      T3.NET= 000020      $ERR57= ***** GX      .CXVEC 000026
PF.ALL= 177777      TS.HLD= 002000      T3.NSD= 000200      $ERR58= ***** GX      .CXVSM 000016
PF.INS= 000040      TS.MSG= 020000      T3.PMD= 040000      $ERR59= ***** GX      .FRKHD= ***** GX
PF.LOG= 000100      TS.NRP= 010000      T3.PRIV= 010000      $ERR60= ***** GX      .HEADR= ***** GX
PF.REQ= 000200      TS.OUT= 000400      T3.REM= 020000      $ERR61= ***** GX      .LLCTB 000660R 002
PS.APR= 000007      TS.RDN= 040000      T3.ROV= 000040      $ERR62= ***** GX      .NETPF= ***** GX
PS.CHK= 010000      TS.RUN= 004000      T3.RST= 000400      $ERR63= ***** GX      .NTGCD 000000
PS.CKP= 040000      TS.STP= 001000      T3.SLV= 002000      $ERR64= ***** GX      .NTPCB 000000
PS.CKR= 020000      T$SKMG= 000000      T3.SWS= 000002      $ERR65= ***** GX      .NTUMR 000002
PS.COM= 000200      T$MIN= 000000      UA.ACC= 000001      $ERR66= ***** GX      .PDVTA= ***** GX
PS.DEL= 000010      T.ACTL 000052      UA.ALL= 000400      $ERR67= ***** GX      .PDVTB 000634R 002
PS.DRV= 000020      T.ASTL 000016      UA.CAL= 00 100      $ERR68= ***** GX      .PLGTH= ***** GX
PS.FXD= 004000      T.ATT 000062      UA.COM= 000200      $ERR9E= ***** GX      .SBPCB 000066
PS.LIO= 001000      T.CPCB 000004      UA.ECH= 000004      $Fmask= ***** GX      .SLTMB 000646R 002
PS.NSF= 000400      T.DPRI 000040      UA.PRC= 000002      $GETRV= ***** GX      .STKDP= ***** GX
PS.QU= 100000      T.EFLG 000022      UA.PUT= 000040      $LLCTA= ***** GX      .TLGTH= ***** GX
PS.PER= 002000      T.IOC 000003      UA.SPE= 000020      $NTLHB= ***** GX      .TSKHD= ***** GX
PS.PIC= 000100      T.LBN 000041      UA.TRA= 002000      $NTLPT= ***** GX
PS.SVS= 000040

```

. ABS. 177776 000 (RW,1,GBL,ABS,OVR)

[illegible]

```

655                                     .SBITL MOVE - Move bytes                ;[TD001]
656                                     +                                         ;[TD001]
657                                     *** MOVE - Move bytes                    ;[TD001]
658                                     INPUTS                                     ;[TD001]
659                                     R0 - Source address                        ;[TD001]
660                                     R1 - Number of bytes to move                ;[TD001]
661                                     R2 - Target address                        ;[TD001]
662                                     OUTPUTS                                     ;[TD001]
663                                     R2 - Address of last byte moved + 1          ;[TD001]
664                                     R0, R1 destroyed                          ;[TD001]
665                                     -                                         ;[TD001]
666                                     MOVE::                                     ;[TD001]
667                                     TST      R1                               ;[TD001]
668                                     BEQ      20$                             ;[TD001]
669 001436                                ; Check for nothing to do              ;[TD001]
670 001436 005701                        ;[TD001]
671 001440 001403                        ;[TD001]
672 001442 112022                        ;[TD001]
673 001444 005301                        ;[TD001]
674 001446 003375                        ;[TD001]
675 001450                                ;[TD001]
                                     10$:  MOVB  (R0)+,(R2)+                    ;[TD001]
                                     DEC      R1                               ;[TD001]
                                     BGT      10$                             ;[TD001]
                                     20$:  RETURN                             ;[TD001]

```

1122
 1123
 1124
 1125
 1126
 1127
 1128
 1129
 1130
 1131
 1132
 1133
 1134
 1135
 1136
 1137
 1138
 1139
 1140
 1141
 1142
 1143
 1144
 1145
 1146
 1147
 1148
 1149
 1150
 1151
 1152
 1153
 1154
 1155
 1156
 1157
 1158
 1159
 1160
 1161
 1162
 1163
 1164
 1165
 1166
 1167
 1168
 1169
 1170
 1171
 1172
 1173
 1174
 1175
 1176
 1177

L 5

```

1122
1123 001700          101$:  EMSG$R  ZQ          ; PROCESS EXTENSION TOO BIG
1124
1125          ;
1126          ; CONTROLLER NUMBER (SLT$DF, CNT$DF)
1127
1128 001706          C.CTL:
1129 001706 126767 000000G 000015' S.CTL:  CMPB      .PNUMB,CTL      ; REJECT TRANSITION IF NO MATCH
1130 001714 001404          BEQ      10$          ;
1131 001716          REJ$          ;
1132 001726          10$:  RETURN          ;
1133
1134          ;
1135          ; UNIT NUMBER (SLT$DF)
1136
1137 001730 126767 000000G 000016' S.UNT:  CMPB      .PNUMB,UNT      ; REJECT TRANSITION IF NO MATCH
1138 001736 001404          BEQ      10$          ;
1139 001740          REJ$          ;
1140 001750          10$:  RETURN          ;
1141
1142          ;
1143          ; NI ROUTER ADJACENCY
1144
1145 001752 016767 000000G 000000G S.BRA:  MOV      .PNUMB,$$NBRA      ; STORE NI ROUTER ADJACENCY VALUE
1146 001760          RETURN          ; LIMIT CHECKED DURING MARK FOR ENABLE SCAN
1147
1148          ;
1149          ; ROUTER PRIORITY VALUE
1150
1151 001762 022767 000177 000000G S.PRI:  CMP      #127, .PNUMB      ; LEGAL VALUE
1152 001770 002404          BLT      10$          ; IF NO - BRANCH
1153 001772 016767 000000G 000000G      MOV      .PNUMB,$$RPRI      ; STORE ROUTER PRIORITY VALUE
1154 002000          RETURN          ;
1155
1156          10$:  REJ$          ; ILLEGAL PRIORITY VALUE
1157 002012          RETURN          ;
1158
1159          ;
1160          ; COUNTER TIMER (SLT$DF)
1161
1162 002014 005767 000000G          S.CTIM:  TST      .PNUMH      ; VALID COUNTER TIMER VALUE?
1163 002020 001004          BNE      101$          ; BR IF NO
1164 002022 016767 000000G 000000G      MOV      .PNUMB,$$CTIM      ; ELSE SAVE COUNTER TIMER VALUE
1165 002030          RETURN          ;
1166 002032          101$:  EMSG$R  YR          ;
1167
1168          ;
1169          ; HELLO/LISTEN TIMER
1170
1171 002040 005767 000000G          S.HTIM:  TST      .PNUMH      ; DOUBLE WORD VALUE
1172 002044 001011          BNE      101$          ; IF YES - BRANCH
1173 002046 016767 000000G 000000G      MOV      .PNUMB,$$HTIM      ; SAVE HELLO TIMER VALUE
1174 002054 006367 000000G          ASL      .PNUMB          ; MULTIPLY BY 2
1175 002060 016767 000000G 000000G      MOV      .PNUMB,$$LTIM      ; SAVE LISTEN TIMER

```

M 5

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
84

[illegible][illegible]

```

VV      VV      CCCCCCCC  FFFFFFFFFF  PPPPPPPP
VV      VV      CCCCCCCC  FFFFFFFFFF  PPPPPPPP
VV      VV      CC        FF          PP          PP
VV      VV      CC        FF          PP          PP
VV      VV      CC        FF          PP          PP
VV      VV      CC        FF          PP          PP
VV      VV      CC        FFFFFFFF  PPPPPPPP
VV      VV      CC        FFFFFFFF  PPPPPPPP
VV      VV      CC        FF          PP
VV      VV      CC        FF          PP
VV      VV      CC        FF          PP
VV      VV      CC        FF          PP
VV      VV      CC        FF          PP
VV      VV      CCCCCCCC  FF          PP
VV      VV      CCCCCCCC  FF          PP

```

```

LL          SSSSSSSS  TTTTTTTTTT
LL          SSSSSSSS  TTTTTTTTTT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SSSSSS    TT
LL          SSSSSS    TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SS        TT
LL          SSSSSSSS  TT
LLLLLLLLLLL SSSSSSSS  TT
LLLLLLLLLLL SSSSSSSS  TT

```

VCFP
Symb

A\$SC
A\$SC
A\$SP
A\$ST
BLKS
BLKS
BLKS
BLKS
CERM
CFER
CFGE
CFGE
CFGN
CFGS
CFGL
CHKL
CHKW
CHNL
CL.C
COUNT
CTIM
CUNN
CVTE
C\$S
C\$S
C\$S
DARD
DARN
DEVN
DEVN
DEVL
DE.C
DSAI
DSAI
DSAI
DSAI
DSAI
DSAI
DSAI
DSCI
DSCI
DSCI
DSFI
DSTI
DSTI
DSTI
DSTI
DSTI
DSTI
DSTI
DSTI
DSTI
DSTI

ASSCHK= 000000	DS.X29= 000002 G	LN.LOO= 000003	PF.XGA= ***** GX	XSSDBT= 000000
ASSCPS= 000000	DTEDES 000370RG	002 LN.OAU= 000003	PORTNO 000354RG	002 Z.NAM= ***** GX
ASSPRI= 000000	DTEFLG 000372RG	002 LN.OFF= 000001	PVSTA 000402RG	002 \$AZFS = ***** GX
ASSTRP= 000000	DSSBUG= 177514	LN.ON = 000000	PV.OFF= 000001 G	\$BFR = ***** GX
BLKSMN= 000040	DSSISK= 200000	LN.OOP= 000004	PV.ON = 000002 G	\$CAT5 = ***** GX
BLKSMX= 002000	DSSLL1= 000001	LN.OPE= 000001	PSSP45= 000000	\$CDTB = ***** GX
BLKSZD 000360RG	DSSYNC= 000000	LN.REF= 000002	PSSWRD= 000000	\$CEACC= ***** GX
BLKSZM 000362RG	DSSYNM= 000000	LN.SER= 000002	QSSOPT= 000010	\$CLBUF= ***** GX
CERR 000534RG	ERRBUF 000022R	002 LN.STA= 000017	RBRA = 000076	\$CLDEQ= ***** GX
CFERR 000544RG	ERRMSG 000012R	002 LN.SUB= 000360	READ 000442R	\$CLFLG= ***** GX
CFGFB 000130RG	ESSXPR= 000000	LN.TRI= 000006	RNACCT 000745RG	002 \$CLIOS= ***** GX
CFGEXT 000006R	FMT10 001772RG	002 LSSASG= 000000	RNAL 000744RG	002 \$CLQUE= ***** GX
CFGNAM 000000R	FMT8 001752RG	002 LSSDRV= 000000	RNETWK 000663RG	002 \$CLSAV= ***** GX
CFGSZ 000336RG	FM.10 = 000000	LSSP11= 000001	RNNL 000703RG	002 \$CLSBK= ***** GX
CFLIN 000636RG	FM.8 = 000000	LSS11R= 000000	RNNODE 000704RG	002 \$CLSTZ= ***** GX
CHKBLK 000742RG	FNDPDV 001202RG	L.COST 000015	NPASS 000734RG	002 \$DFUIC= ***** GX
CHKWND 001000RG	FNLIN 000632RG	L.CTL 000012	RNPL 000733RG	002 \$ERRMA 001522RG
CHNLMX= 007777 G	F\$LVL= 000001	L.CVA 177776	RNTL 000662RG	002 \$ERRMB 001552RG
CL.OPN= ***** GX	F.RSIZ= ***** GX	L.DDM 000002	RNUL 000712RG	002 \$ERRM4 001412RG
COUNT 000356RG	G\$STPP= 000000	L.DDS 000004	RNUER 000713RG	002 \$ERRM5 001442RG
CTIM 000400RG	G\$STSS= 000000	L.DLC 000003	RNWND= ***** GX	002 \$ERRM6 001472RG
CUNMMX= 000006 G	G\$STTK= 000000	L.DLM 000006	RTSPC 000630RG	002 \$ERRR0 001112RG
CVTBUF 000357RG	G\$SWRD= 000000	L.DLS 000010	R\$SDER= 000000	002 \$ERRR1 001146RG
C\$CKP= 000000	HSHADD 000376RG	002 L.FLG 000000	R\$SKT1= 000001	002 \$ERRR2 001222RG
C\$SORE= 000400	HSHSZ 000374RG	002 L.KRBA 000016	R\$SND= 000000	002 \$ERRR5 001276RG
C\$SRSH= 177564	IE.EOF= ***** GX	L.LEN = 000022	R\$S11M= 000000	002 \$ERRR6 001332RG
DARDEN= 000516RG	002 IE.RBG= ***** GX	L.MPF 000022	SF.ACT= 000200	002 \$ERRR9 001352RG
DARMLN= 000017 G	I\$RAR= 000000	L.NMST 000020	SF.ENA= 000100	002 \$ERRP7 000766RG
DEVCTL 000700RG	I\$SRDN= 000000	L.NSTA 000014	SF.LPB= 000004	002 \$ERRP8 001022RG
DEVNAM 000652RG	K\$SCNT= 177546	L.OWNR 000021	SF.MFL= 000040	002 \$ERRP9 001056RG
DEVUNT 000716RG	K\$SCSR= 177546	L.UNT 000013	SF.PAC= 000020	002 \$ERR1P 001602RG
DE.OFF= 000002 G	K\$SLDC= 000000	MAXID = 000020	SF.REA= 000010	002 \$ERR1Q 001640RG
DE.ON = 000001 G	K\$STPS= 000074	MOVE 001430RG	SF.SER= 000001	002 \$ERR1R 001676RG
DSACGL 000533RG	LBRA = 000074	MRKFL 000344RG	SF.SVC= 000002	002 \$ERR1T 001724RG
DSACUG 000534RG	LD\$LP = 000000	MXACAC= 000020	SF.UNL= 000040	\$FNOUT= ***** GX
DSAHCT 000521RG	LF.ACT= 100000	MXACPA= 000010	SPACE = 000040	\$GCLEP= ***** GX
DSALCT 000520RG	LF.BRO= 000400	MXACUS= 000020	SPFILL 001444RG	\$GETAD= ***** GX
DSARCT 000522RG	LF.BWT= 000007	MXNOID= 000006	SPSAV 000340R	002 \$GETRV= ***** GX
DSARDP 000523RG	LF.ENA= 002000	M\$SCRB= 000124	STORID 001324RG	\$NITCFP 000000RG
DSARDT 000500RG	LF.LPB= 001000	M\$SCRX= 000000	STRNXT 001254RG	\$PDVNM= ***** GX
DSASHI 000556RG	LF.MDC= 000100	M\$SFCS= 000000	SUBAMN= 000000 G	\$PDVTA= ***** GX
DSASLO 000554RG	LF.MFL= 004000	M\$SMGE= 000000	SUBAMX= 023417 G	\$PFLAG= ***** GX
DSCMCT 000560RG	LF.MTP= 000020	M\$SNET= 000000	SYNERR 000342RG	002 \$QCHN= ***** GX
DSCMSK 000561RG	LF.PAC= 000200	M\$SOVR= 000000	SYSFDB= ***** GX	\$QCUC= ***** GX
DSCVAL 000622RG	LF.RDY= 040000	NEXT 000352RG	S\$SWRG= 000000	\$QDSA = ***** GX
DSCVCT 000621RG	LF.REA= 010000	N\$SACC= 000001	S\$SYSZ= 007600	\$QDSC = ***** GX
DSFLG 000404RG	LF.SER= 000040	N\$SBUF= 000001	S.COST 000001	\$QDSN = ***** GX
DSTEND= ***** GX	LF.TJM= 000010	N\$SLDV= 000001	S.FLG 000000	\$QDST = ***** GX
DSTNAM 000430RG	LF.UNL= 020000	N\$SMCP= 000001	S.LEN 000004	\$QDTE = ***** GX
DSTNML 000427RG	LF.X2P= 000000	N\$SMLL= 000001	S.NMST 000002	\$QPSN = ***** GX
DSTOBJ 000451RG	LINCTL 000350RG	002 N\$SMOV= 000010	S.OWNR 000003	\$QPPVC = ***** GX
DSTPRI 000450RG	LINNAM 000346RG	002 N\$SNCT= 000001	T\$SKMG= 000000	\$QRDT = ***** GX
DSTTKL 000406RG	LINUNT 000351RG	002 N\$SPEM= 000001	T\$SMIN= 000000	\$QRNA = ***** GX
DSTTSK 000407RG	LNKEND 001112RG	OPEN 000400R	V\$SCTR= 001000	\$QRNW = ***** GX
DSTVR0 000452RG	LN.CLO= 000000	PCKBCD 001032RG	WNSDMX= 000177	\$QX29 = ***** GX
DSTVR1 000470RG	LN.DUM= 000005	PF.PSV= ***** GX	WNSDZD 000364RG	002 \$QX3P = ***** GX
DS.X25= 000001 G	LN.LOA= 000004	PF.XDF= ***** GX	WNSDZM 000366RG	002 \$RADDR= ***** GX

```

171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189 000076 012700 000000' PR.LBL: MOV #DSKBUF,R0 ; GET BUFFER ADDRESS
190 000102 CALL $RLBL ; READ LABEL BLOCK
191 000106 103441 BCS 101$ ; IF CS, FAILURE
192 000110 032760 040000 000030 BIT #T$NHD,L$BFLG(R0) ; DOES CETAB HAVE A HEADER?
193 000116 001444 BEQ 111$ ; IF EQ, YES
194 000120 016001 000016 MOV L$BLDZ(R0),R1 ; GET LOAD SIZE IN BLOCKS
195 000124 026001 000020 CMP L$BMXZ(R0),R1 ; IS CETAB OVERLAID?
196 000130 001037 BNE 111$ ; IF NE, YES
197 000132 022701 000200 CMP #200,R1 ; IS CETAB > 4K WORDS?
198 000136 103444 BLO 131$ ; IF LO, YES
199 000140 016002 000352 MOV L$BEXT(R0),R2 ; GET EXTEND SIZE
200 000144 060102 ADD R1,R2 ; ADD CETAB SIZE TO EXTEND SIZE
201 000146 022702 000200 CMP #200,R2 ; IS CETAB+EXTENSION > 4K WORDS?
202 000152 103442 BLO 141$ ; IF LO, YES
203 000154 ASL$ 6,R1 ; CONVERT FROM BLOCKS TO BYTES
204 000170 010167 000000G MOV R1,$LLEN ; SET THE TRANSFER SIZE
205 000174 005002 CLR R2 ; SUCCESS INDICATOR
206
207 000176 CALL $ALL16 ; TRY TO ALLOCATE ROOM FOR CETAB
208 000202 103416 BCS 121$ ; IF CS, COULD NOT
209 000204 010102 MOV R1,R2 ; COPY THE CETAB LENGTH
210 000206 010001 MOV R0,R1 ; COPY THE CETAB ADDRESS
211 000210 RETURN ; RETURN TO CALLER
212
213 ; ERROR CONDITIONS
214
215 000212 116777 000000G 000000G 101$: MOVB $IOSB,$CLIOS ; COPY ERROR CODE
216 000220 012767 000010 000000' MOV #ERR8,ERROR ; LABEL BLOCK READ FAILURE
217 000226 000417 BR 142$
218 000230 012767 000012 000000' 111$: MOV #ERR10,ERROR ; INVALID CEX IMAGE
219 000236 000413 BR 142$
220 000240 012767 000044 000000' 121$: MOV #ERR36,ERROR ; CETAB ALLOCATION FAILURE
221 000246 000407 BR 142$
222 000250 012767 000016 000000' 131$: MOV #ERR14,ERROR ; CEX EXCEEDS 4K WORDS
223 000256 000403 BR 142$
224 000260 012767 000020 000000' 141$: MOV #ERR16,ERROR ; CEX EXTENSION EXCEEDS 4K WORDS
225 000266 000261 142$: SEC ; INDICATE FAILURE
226 000270 RETURN

```

VCLQR
CLQRE

VCLQRM - REMOVE ENTRIES FROM CL MACRO V05.03b Saturday 29-Jun-85 02:18 Page 7-1
 CLQREM - REMOVE ENTRIES FROM CLOCK QUEUE

126	000200	011402	MOV	(R4),R2	; GET ADDRESS OF NEXT ENTRY
127	000202		GETRV	R3	; READ IN PREVIOUS ENTRY
128	000212	010214	MOV	R2,(R4)	; UN-LINK ENTRY FROM LIST
129	000214		PUTRC		; REWRITE CLOCK QUEUE ENTRY
130					
131	000220	020526	CMP	R5,(SP)+	; NITNIT TCB ADDRESS ?
132	000222	001004	BNE	17\$; IF NOT - MUST BE AUX CLOCK BLOCK
133	000224	012701 000020	MOV	#C.LGTH,R1	; GET LENGTH OF CLOCK QUEUE ENTRY
134	000230		CALL	\$DEA16	; DE-ALLOCATE ENTRY BLOCK
135					
136	000234	010300	17\$: MOV	R3,R0	; PREVIOUS ENTRY IS NOW CURRENT ENTRY
137	000236	000705	BR	10\$; AND LOOK FOR NEXT ENTRY
138					
139	000240		20\$: RETURN		

VDEV
LOCA

```

64          .SBTTL  LOCAL DATA
65
66          **      LOCAL DATA
67          **
68
69 000000          .PSECT  DATA,D
70
71          .NLIST  BEX
72
73          ;
74          ; SYMBOL TABLE EXTENSION
75          ;
76 000000      123      124      102  STBEXT: .ASCIZ  'STB'
77
78          ;
79          ; DEVICE NAME PRECEDED BY '$'
80
81 000004      044      130      130  DNAME: .ASCII  '$XX'
82
83          ;
84          ; ERROR MESSAGE BUFFER
85
86 000007      015      120  ERRMSG: .BYTE   15
87 000010      126      120  .ASCII  'VNP -- '
88 000017      126      120  ERRBUF: .BLKB   70
89
90          .EVEN
91
92          ;
93          ; SYMBOL LOOKUP CONTROL BLOCKS
94
95 000126  000140'  000000  DBLK1: .WORD   DBLK2,0          ; THIS MUST BE THE FIRST CONTROL BLOCK
96 000132  126230  020564  .RAD50  '$XEND'
97 000136  000000  .LENGTH: .WORD   0
98 000140  000000  000000  DBLK2: .WORD   0,0
99 000144  126230  014572  .RAD50  '$XXDCB'
100 000150  000000  DCB:      .WORD   0
101
102          ;
103          ; DCB DISPATCH TABLE NAME (THIS SYMBOL COMES FROM THE PROCESS STB FILE)
104
105 000152  126230  076534  DUMMY: .RAD50  '$XTBL'
106
107          ;
108          ; STB FILE ERRORS
109
110 000156  002226'  ETAB:   .WORD   $ERR2B          ; OPEN FAILURE
111 000160  002264'  .WORD   $ERR2C          ; READ FAILURE
112 000162  002322'  .WORD   $ERR2D          ; READ OVERFLOW
113 000164  002350'  .WORD   $ERR2E          ; BAD STB FORMAT
114
115          ;
116          ; FLAG WHETHER OR NOT DEVICE STRUCTURES ALREADY EXIST
117
118 000166          FOUND: .BLKW   1
119
120          ;

```


VDEV - VNP LOAD DEVICE STRUCTUR MACRO V05.03b Tuesday 03-Sep-85 10:28 ^{L 13} Page 13-3
DVXFR - TRANSFER TABLES INTO CORE

636 002050 006026
637 002052

ROR (SP)+
RETURN

; RESTORE C-BIT

VD
Ta

VDIS - VNP SHOW COMMANDS
Table of contents

MACRO V05.03b Monday 15-Jul-85 L 14
12:14

62- 4831	FNDUCB - GET UCB OF NS: DEVICE
63- 4858	CUNDEC - CONVERT CUG NUMBER TO DECIMAL
64- 4914	BCDASC - CONVERT BCD TO ASCII
65- 4951	HEXASC - CONVERT HEX TO ASCII
66- 4986	MSGOUT - OUTPUT MESSAGE TO TI:

VDIS - VNP SHOW COMMANDS
\$DALI - SHOW ALIAS

MACRO V05.03b Monday 15-Jul-85 12:14^{L 15} Page 10-2

553

,DSABL LC

1059	005616	010167	173734	MOV	R1,TEMP7	; STORE AREA NUMBER FOR OUTPUT
1060						
1061	005622			ERRPT\$	L0MS3X,MSGOUT	; PRINT NODE AREA.ADDRESS
1062	005632	000421		BR	90\$; CONTINUE
1063						
1064	005634					
1065	005634	000241	70\$:	CLC		; DO NOT ROR C-BIT
1066		000012		.REPT	10.	
1067				ROR	R1	; GET AREA NUMBER
1068				.ENDM		
1069	005662	010167	173670	MOV	R1,TEMP7	; STORE AREA NUMBER FOR OUTPUT
1070						
1071	005666			ERRPT\$	L0MS3Y,MSGOUT	; PRINT NODE AREA.ADDRESS
1072						
1073	005676		90\$:	RETURN		

```

453 000600      CALL    $AZFS      ;
454 000604 012700 000634'    MOV    #PDVTB,R0    ; GET FIRST LOOKUP CONTROL BLOCK ADDRESS
455 000610      CALL    $SYMBL    ; LOOK FOR SYMBOLS IN STB FILE
456 000614 103421    BCS     STBERR ; IF CS, FILE ERROR
457
458      .IF DF R$$MPL
459
460      MOV    #SDFUIC,R0    ; SETUP ASCII FILE STRING
461      MOV    #VEXNAM,R1    ; FOR [XXX,R4]RSXVEC.STB
462      MOV    #STBEXT,R2    ; ...
463      CALL    $AZFS      ;
464      MOV    #VXBLK1,R0    ; GET LIST OF SYMBOLS
465      CALL    $SYMBL    ; DEFINE SYMBOLS
466      BCS     STBERR      ; IF ERROR - BRANCH
467
468      SWITCH #TMPSTR      ; USE TEMPORARY OUTPUT BUFFER
469      MOV    #10,TMPSTR   ; FUNCTION CODE TO DISPATCH 100 MSEC TIMEOUTS
470      PUTRC .STMFC,#2     ; FILE IT
471      MOV    #1010,TMPSTR ; FUNCTION CODE TO DISPATCH LONG TIMEOUTS
472      PUTRC .LTMFC       ; FILE IT
473      CLR    TMPSTR      ; MAKE A NULL
474      PUTRC .CCBAF       ; INITIALIZE ALLOCATION FAILURES
475      PUTRC .RDBAF       ; ...
476      PUTRC .LDBAF       ; ...
477      PUTRC .SDBAF       ; ...
478      PUTRC .CCBAL       ; INITIALIZE NUMBER OF DYNAMIC CCB'S
479      MOV    .CMFRK,R1    ; ADDRESS OF FORK BLOCK
480      ADD    #4,R1        ; POINT AT LIST HEAD
481      PUTRC  R1           ; INITIALIZE FORK BLOCK LISTHEAD
482      MOV    R1,TMPSTR    ; ...
483      ADD    #2,R1        ; ...
484      PUTRC  R1           ;
485      SWITCH             ; GO BACK TO DEFAULT OUTPUT BUFFER
486
487      .IFF
488
489 000616 012700 000434'    MOV    #CBLK1,R0    ; START AT FIRST SYMBOL BLOCK
490 000622 005760 000002    10$: TST     2(R0)      ; IS SYMBOL DEFINED?
491 000626 001404          BEQ     20$      ; IF EQ, NO - DON'T WORRY
492 000630 026070 000010 000012 CMP     10(R0),a12(R0) ; DOES VALUE MATCH TKB DEFINITION?
493 000636 001005          BNE     101$     ; IF NE, NO - VERY BAD
494 000640 011000          20$: MOV     (R0),R0    ; GET NEXT SYMBOL BLOCK ADDRESS
495 000642 022700 000514'    CMP     #XBLK1,R0    ; IS IT THE FIRST PROCESS VERIFICATION BLOCK?
496 000646 001365          BNE     10$      ; IF NE, NOT AT END OF LIST
497
498      .ENDC
499
500 000650      RETURN
501
502
503      ; ERROR CONDITIONS
504
505 000652      101$: MSG$R 48      ; CEX/RSX INCOMPATIBLE
506 000650 016000 000714'    STBERR: MOV    ETAB1-2(R0),R0 ; GET ERROR MESSAGE CONTROL BLOCK ADDRESS
507 000654      CALLR  a(R0)+    ; PRINT ERROR MESSAGE
508 000654

```

VCEX
Symbol

DATA
Error

*** A

Work
Work
Size
Size
Opera

Elaps
SY:VC


```

162 000160 000113 .WORD 75.
163 000162 000156 .WORD 110.
164 000164 000206 .WORD 134.
165 000166 000226 .WORD 150.
166 000170 000454 .WORD 300.
167 000172 001130 .WORD 600.
168 000174 002260 .WORD 1200.
169 000176 003410 .WORD 1800.
170 000200 003720 .WORD 2000.
171 000202 004540 .WORD 2400.
172 000204 007020 .WORD 3600.
173 000206 011300 .WORD 4800.
174 000210 016040 .WORD 7200.
175 000212 022600 .WORD 9600.
176 000214 022600 .WORD 9600.
177
178 ; ERROR MESSAGES
179 ;
180 000216 NTLERS $1A,8B,CERR,RTSPC,CFLIN,<station number.>
181 000246 NTLERS $1B,8,CERR,RTSPC,CFLIN,<More than 48. STA$DF'S.>
182 000306 NTLERS $1C,8B,CERR,RTSPC,CFLIN,<interrupt vector.>
183 000340 NTLERS $1D,8B,CERR,RTSPC,CFLIN,<Vector not in system.>
184 000376 NTLERS $1E,8B,CERR,RTSPC,CFLIN,<CSR address.>
185 000424 NTLERS $1F,8B,CERR,RTSPC,CFLIN,<interrupt priority.>
186 000460 NTLERS $1G,8B,CERR,RTSPC,CFLIN,<unit number.>
187 000506 NTLERS $1H,8,CERR,RTSPC,CFLIN,<UNT$DF out of order.>
188 000544 NTLERS $1J,8B,CFERR,RTSPC,CFLIN,<logical link count.>
189 000600 NTLERS $1K,8B,CERR,RTSPC,CFLIN,<characteristics value.>
190 000640 NTLERS $1L,8B,CERR,RTSPC,CFLIN,<secondary CSR address.>
191 000700 NTLERS $1M,8,CERR,RTSPC,CFLIN,<Secondary CSR not allowed.>
192 000744 NTLERS $1N,8,CERR,RTSPC,CFLIN,<Secondary CSR missing.>
193 001004 NTLERS $1P,8,CFERR,$REP.N,<Open failure (-***).>
194 001042 NTLERS $1Q,8,CFERR,$REP.N,<Read failure (-***).>
195 001100 NTLERS $1R,8,CFERR,RTSPC,CFLIN,<Read overflow.>
196 001130 NTLERS $1S,8,CERR,RTSPC,CFLIN,<UNT$DF missing.>
197 001160 NTLERS $<,>1I,8,CERR,RTSPC,CFLIN,<Syntax error>
198 001206 NTLERS $1U,8,CFERR,RTSPC,<<SLT$DF missing ( EOF ).>
199 001246 NTLERS $1V,8,CFERR,RTSPC,<<CNT$DF missing ( EOF ).>
200 001306 NTLERS $1W,8,CFERR,RTSPC,<<UNT$DF missing ( EOF ).>
201 001346 NTLERS $1X,8,CFERR,RTSPC,<<STA$DF missing ( EOF ).>
202 001406 NTLERS $1Y,8,CFERR,RTSPC,<<DDM$DF/DLC$DF missing ( EOF ).>
203 001456 NTLERS $1Z,8,CFERR,RTSPC,<<LLC$DF missing ( EOF ).>
204 001516 NTLERS $2A,8B,CERR,RTSPC,CFLIN,<Local physical station address>
205 001566 NTLERS $2B,8,CERR,RTSPC,CFLIN,< Illegal PLT$DF parameter>
206 001630 NTLERS $2Q,8,CERR,RTSPC,CFLIN,<LLC$DF Process Extension too Big>
207 001702 NTLERS $YA,8B,CERR,RTSPC,CFLIN,<hello timer value>
208 001734 NTLERS $YB,8B,CERR,RTSPC,CFLIN,<listen timer value>
209 001770 NTLERS $YP,8B,CERR,RTSPC,CFLIN,<line cost>
210 002012 NTLERS $YQ,8B,CERR,RTSPC,CFLIN,<Multipoint Active Ratio>
211 002052 NTLERS $YR,8B,CERR,RTSPC,CFLIN,<counter timer value>
212 002106 NTLERS $YS,8B,CERR,RTSPC,CFLIN,<maximum non-counter value>
213 002152 NTLERS $3A,8,CFERR,RTSPC,<<X3P$DF missing (EOF)>
214 002210 NTLERS $3B,8,CFERR,RTSPC,<<X2P$DF missing (EOF)>
215 002246 NTLERS $3C,8,CFERR,RTSPC,<<SVC$DF/PV$DF missing (EOF)>
216 002312 NTLERS $<,>2C,8B,CERR,RTSPC,CFLIN,<maximum block size>
217 002346 NTLERS $<,>2F,8B,CERR,RTSPC,CFLIN,<maximum window size>
218 002402 NTLERS $<,>2I,8B,CERR,RTSPC,CFLIN,<line-id>

```

```

677          .SBTTL  SPFILL - Fill area with spaces          ;[TD001]
678          ;+                                             ;[TD001]
679          ;*** SPFILL - Fill area with spaces           ;[TD001]
680          ; INPUTS                                       ;[TD001]
681          ;     R0 - address of area to fill             ;[TD001]
682          ;     R1 - length of area to fill             ;[TD001]
683          ; OUTPUTS                                       ;[TD001]
684          ;     All registers preserved                  ;[TD001]
685          ; -                                             ;[TD001]
686          SPFILL::                                       ;[TD001]
687          MOV     R0,-(SP)                                ; save registers ;[TD001]
688          MOV     R1,-(SP)                                ;[TD001]
689          BEQ     20$                                     ; nothing to do? ;[TD001]
690          MOV     R0,#SPACE,(R0)+                        ;[TD001]
691          10$:    MOV     R1,10$                          ;[TD001]
692          DEC     R1                                      ;[TD001]
693          BGT     10$                                     ;[TD001]
694          20$:    MOV     (SP)+,R1                        ; restore regs ;[TD001]
695          MOV     (SP)+,R0                                ;[TD001]
696          RETURN                                         ;[TD001]
697
001452
001452 010046
001454 010146
001456 001404
001460 112720 000040
001464 005301
001466 003374
001470 012601
001472 012600
001474
    
```

1179
 1180
 1181
 1182
 1183
 1184
 1185
 1186
 1187
 1188
 1189
 1190
 1191
 1192
 1193
 1194
 1195
 1196
 1197
 1198
 1199
 1200
 1201
 1202
 1203
 1204
 1205
 1206
 1207
 1208
 1209
 1210
 1211
 1212
 1213
 1214
 1215
 1216
 1217
 1218
 1219
 1220
 1221
 1222
 1223
 1224
 1225
 1226
 1227
 1228
 1229
 1230
 1231
 1232
 1233
 1234
 1235

VCFG -
FNPDV

```

1656
1657
1658
1659 004172 005767 000000G
1660 004176 001012
1661 004200 016700 000000G
1662 004204 022700 000040
1663 004210 101005
1664 004212 022700 002000
1665 004216 103402
1666 004220 000241
1667 004222 000401
1668 004224 000261
1669 004226
1670
1671
1672
1673 004230 005767 000000G
1674 004234 001010
1675 004236 016700 000000G
1676 004242 002405
1677 004244 022700 000177
1678 004250 103402
1679 004252 000241
1680 004254 000401
1681 004256 000261
1682 004260
1683
1684 000001

; CHECK FOR LEGAL BLOCK SIZE VALUE
CHKBLK::TST ,PNUMH ; DOUBLE WORD VALUE?
; BNE 10$ ; BR IF YES - ERROR
MOV ,PNUMB,R0 ; GET BLOCK SIZE VALUE
CMP #BLKSMN,R0 ; IS BLOCK SIZE WITHIN RANGE?
; BHI 10$ ; BR IF NO - ERROR
CMP #BLKSMX,R0 ;
; BLO 10$ ; BR IF NO - ERROR
CLC ; INDICATE LEGAL VALUE
BR 20$ ; AND RETURN
10$: SEC ; INDICATE ILLEGAL VALUE
20$: RETURN

; CHECK FOR LEGAL WINDOW SIZE VALUE
CHKWND::TST ,PNUMH ; DOUBLE WORD VALUE?
; BNE 10$ ; BR IF YES - ERROR
MOV ,PNUMB,R0 ; GET WINDOW BLOCK VALUE
BLT 10$ ; BR IF ILLEGAL VALUE
CMP #WINDSMX,R0 ; WITHIN VALID RANGE?
; BLO 10$ ; BR IF NO - ERROR
CLC ; INDICATE LEGAL VALUE
BR 20$ ; AND RETURN
10$: SEC ; INDICATE ILLEGAL VALUE
20$: RETURN

.END
```

5-	59	MACRO DEFINITIONS	
6-	80	CONSTANT DEFINITIONS	
7-	102	LOCAL DATA	
8-	254	ERROR MESSAGES	
9-	289	\$NTICFP - CONFIG FILE SCAN	
10-	409	OPEN CONFIG FILE	
11-	433	READ - READ NEXT RECORD	
12-	464	CERR,CFERR - PRINT ERROR	
13-	503	CFLIN - ECHO LAST RECORD	
14-	518	COMMON ACTION ROUTINES - LINE-ID	
15-	577	PCKBCD - PACK STRING OF DIGITS IN BCD FORMAT	
16-	614	LNKEND - LINK BLOCK AT END OF LIST	
17-	640	FNDPDV - FIND PDV	
18-	672	STRNXT - STORE NEXT DIGIT	: [TD001]
19-	698	STORID - Store id-string according to system type	: [TD001]
20-	761	MOVE - Move bytes	: [TD001]
21-	783	SPFILL - Fill area with spaces	

```

464                                     .SBTTL CERR,CFERR - PRINT ERROR
465
466                                     *
467 CERR - PRINT ERROR MESSAGE
468 CFERR - PRINT FILE ASSOCIATED ERROR MESSAGE
469
470 INPUTS:
471     R0=MESSAGE BLOCK ADDRESS
472
473 OUTPUTS:
474     C-BIT=SET
475     RETURN TO $NLCFG'S CALLER
476
477 000534 010046 CERR:: MOV R0,-(SP) ; SAVE R0
478 000536      CALL $CLDEQ ; CLOSE THE CONFIG FILE
479 000542      MOV (SP)+,R0 ; RESTORE R0
480 000544      MOV R0,R5 ; COPY ERP MESSAGE BLOCK ADDRESS
481 000546      MOV #ERRBUF,R4 ; PLACE TO BUILD ERROR MESSAGE
482 000552      MOV (R5)+,R3 ; GET FORMAT STRING ADDRESS
483 000554      CALL @R5+ ; DO THE FORMATTING
484 000560      TSTB (R3) ; IS IT END OF FORMAT STRING?
485 000562      BEQ 20$ ; IF EQ, YES
486 000564      MOVB (R3)+,(R4)+ ; ELSE COPY REST OF FORMAT STRING
487 000566      BNE 10$
488 000570      DEC R4 ; SKIP THE ZERO BYTE
489 000572      MOV (R5)+,R3 ; GET LINE 2 ADDRESS
490 000574      MOVB (R5)+,(R4)+ ; COPY THE ERROR TEXT
491 000576      BNE 30$
492 000578      DEC R4 ; SKIP THE ZERO BYTE
493 000600      MOV #ERRMSG,R0 ; GET ERROR MESSAGE ADDRESS
494 000602      MOV R4,R1 ; COPY END ADDRESS
495 000604      SUB R0,R1 ; GIVING LENGTH
496 000610      CALL $TIOUT ; PRINT IT
497 000614      TST R3 ; SHOULD SECOND LINE BE PRINTED?
498 000616      BEQ 40$ ; IF EQ, NO
499 000622      CALL (R3) ; PRINT SECOND ERROR MESSAGE LINE
500 000624      SEC ; RETURN FROM $NLCFG WITH C-BIT SET
501 000630      MOV SPSAV,SP
502
503 RTSPC:: RETURN

```

VCFP
 Symt
 \$REF
 \$TIC
 . AE
 DATA
 Error

 Work
 Work
 Size
 Size
 Oper
 Elap
 SY:V

VCFP - VNP PSI CONFIGURATION FI MACRO V05.03b Saturday 29-Jun-85 M 9 02:17 Page 21-2
Symbol table

\$REP.N= ***** GX \$XLINK= ***** GX .PNUMB= ***** GX .PSTCN= ***** GX .PSTPT= ***** GX
\$TIOUT= ***** GX .PCHAR= ***** GX .PNUMH= ***** GX

. ABS. 177776 000 (RW,I,GBL,ABS,OVR)
 001470 001 (RW,I,LCL,REL,CON)
DATA 002022 002 (RW,D,LCL,REL,CON)
Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 14001 Words (55 Pages)
Size of core pool: 15496 Words (59 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:00:35.05
SY:VCFP:V2,[132,134]VCFP/CR/-SP=SY:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[13: 10]VCFP

```

228      ;
229      ; PR.XFR - READ PROCESS TASK IMAGE INTO CORE
230      ;
231      ; THIS ROUTINE IS CALLED TO READ THE PROCESS TASK IMAGE INTO CORE.
232      ;
233      ; INPUTS:
234      ; R1=START ADDRESS OF WHERE TO LOAD THE CETAB TASK IMAGE
235      ;
236      ; OUTPUTS:
237      ; C-BIT=SUCCESS/FAILURE
238      ;
239      PR.XFR:
240      JSR    R2,$$AVVR      ; SAVE VOLATILE REGISTERS
241      MOV    $LLEN,R2      ; GET LENGTH OF TASK IMAGE
242      MOV    #512,$LLEN    ; TRANSFER 1 BLOCK AT A TIME
243      MOV    #512,R0       ; READ IN BUFFER
244      SUB    #512,R2       ; ATTEMPT TO TRANSFER AT LEAST
245      BGE    20$           ; AT LEAST 512 BYTES.
246      ADD    R2,$LLEN      ; ELSE, TRANSFER REMAINDER
247      CALL   $READ        ; READ INTO MEMORY
248      BCS    10$          ; IF CS, FAILURE
249      ADD    #1,$LBN+2     ; INCREMENT BLOCK NUMBER
250      ADC    $LBN         ; DOUBLE WORD
251      PUTRC  R1,$LLEN      ; WRITE BLOCK TO SYSTEM IMAGE
252      ADD    $LLEN,R1      ; ADJUST POINTER FOR NEXT TIME
253      TST    R2           ; ANY MORE TO TRANSFER ?
254      BGT    10$          ; IF GT, YES
255      CLC                ; ELSE INDICATE SUCCESS
256      RETURN
257      ;
258      ; ERROR CONDITION
259      ;
260      10$:  MOVB    $IOSB,$$CLIOS ; COPY ERROR CODE
261      MOV     $ERR34,$ERR34 ; PROCESS IMAGE READ FAILURE
262      RETURN
263

```

VCLQRM - REMOVE ENTRIES FROM CL MACRO V05.03b Saturday 29-Jun-85 02:18 Page 8
 FNDDPDV - FIND PDV ADDRESS

M.11

.SBTTL FNDDPDV - FIND PDV ADDRESS

```

141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159 000242 010046
160 000244 016701 000000G
161 000250 016702 000000G
162 000254 001407
163 000256 012100
164 000260 001403
165 000262 026016 000000G
166 000266 001404
167 000270 005302
168 000272 003371
169 000274 000261
170 000276 000401
171 000300 000241
172 000302 005726
173 000304
174 000001

      :+
      : FNDDPDV - FIND PDV ADDRESS
      : INPUTS:
      :   RO - RAD50 PROCESS NAME
      : OUTPUTS:
      :   CARRY CLEAR:
      :   RO - PDV ADDRESS OF PROCESS
      :   CARRY SET:
      :   PROCESS NOT FOUND
      :   R1,R2 DESTROYED
      :-

FNDDPDV: MOV    RO,-(SP)          ; SAVE NAME TO LOOK FOR
          MOV    $PDVTA,R1      ; GET PDV TABLE ADDRESS
          MOV    $PDVNM,R2      ; GET NUMBER OF PDV'S
          BEQ    30$            ; BR IF NONE
10$:      MOV    (R1)+,RO        ; GET PDV ADDRESS
          BEQ    20$            ; IF ZERO, SKIP IT
15$:      CMP    Z,NAM(RO),(SP)  ; DOES THE NAME MATCH?
          BEQ    40$            ; BR IF YES - SUCCESS
20$:      DEC    R2             ; DECREMENT COUNT
          BGT    10$            ; BR IF MORE
30$:      SEC                     ; INDICATE PROCESS NOT FOUND
          BR     50$
40$:      CLC                     ; INDICATE SUCCESS
50$:      TST    (SP)+          ; CLEAN UP STACK
          RETURN
          .END

```

VDEV
LOCAL

VDEV - VNP LOAD DEVICE STRUCTOR MACRO V05.03b Tuesday 03-Sep-85 10:28 Page 6-1
 LOCAL DATA

```

121      ; TKB BASE ADDRESS OF XXXTAB.TSK FILE
122
123 000170      BASE: .BLKW 1
124
125      ;
126      ; NUMBER OF UNITS
127
128 000172      UNITS: .BLKW 1
129
130      ;
131      ; SCOM DEVICE DATA STRUCTURES DESCRIPTORS
132
133 000174      PBUF: .BLKW      ; SCOM BUFFER DESCRIPTOR OF LOADABLE
134 000176      PLEN: .BLKW      ; DEVICE DATA STRUCTURES.
135 000200      VREL: .BLKW      ; RELOCATION DIFFERENTIAL FOR BUFFERING
136
137      ;
138      ; END OF TABLES ADDRESS
139
140 000202      TABEND: .BLKW 1
141
142 000204      DEVBUF: .BLKB 2000      ; DEVICE DATA STRUCTURES BUFFER

```

VDEV
 DVER

VDEV - VNP LOAD DEVICE STRUCTUR MACRO V05.03b Tuesday 03-Sep-85 10:28 Page 14
 DVERR - PRINT ERROR

```

639          .SBTTL  DVERR - PRINT ERROR
640      ;+
641      ; DVERR - PRINT ERROR MESSAGE
642      ;
643      ; INPUTS:
644      ; R0=MESSAGE BLOCK ADDRESS
645      ;
646      ; OUTPUTS:
647      ; C-BIT=SET
648      ; -
649
650 002054      DVERR:  CALL    $SAVAL      ; SAVE ALL REGISTERS
651 002060      MOV     R0,R5             ; COPY ERROR MESSAGE BLOCK ADDRESS
652 002062      MOV     #ERRBUF,R4        ; PLACE TO BUILD ERROR MESSAGE
653 002066      MOV     (R5)+,R3         ; GET FORMAT STRING ADDRESS
654 002070      CALL    @ (R5)+          ; DO THE FORMATTING
655 002072      TSTB    (R3)             ; IS IT END OF FORMAT STRING?
656 002074      BEQ     20$              ; IF EQ, NO
657 002076      MOVB    (R3)+,(R4)+      ; COPY THE REST OF THE FORMAT STRING
658 002100      BNE     10$              ;
659 002102      DEC     R4               ;
660 002104      MOV     (R5)+,R3         ; SKIP THE ZERO BYTE
661 002106      MOVB    (R5)+,(R4)+      ; GET LINE 2 ADDRESS
662 002110      BNE     30$              ; COPY THE ERROR TEXT
663 002112      DEC     R4               ;
664 002114      MOV     #ERRMSG,R0       ; SKIP THE ZERO BYTE
665 002120      MOV     R4,R1            ; PRINT THE ERROR MESSAGE
666 002122      SUB     R0,R1            ;
667 002124      CALL    $TIOUT          ;
668 002130      TST     R3               ; IS THERE A LINE 2 ADDRESS?
669 002132      BEQ     40$              ; IF EQ, NO
670 002134      CALL    (R3)            ; GO PRINT SECOND LINE
671 002136      SEC                     ; RETURN C-BIT SET
672 002140      RETURN

```

.TITLE VDIS - VNP SHOW COMMANDS
.IDENT /V05.00/
.ENABL LC
.NLIST BEX

.*
: COPYRIGHT (C) 1979, 1980, 1982, 1983, 1985 BY
: DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

: THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A
: SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE
: INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR
: ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE
: MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH
: SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE
: TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN
: IN DEC.

: THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT
: NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
: EQUIPMENT CORPORATION.

: DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF
: ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

: MODULE DESCRIPTION:

: VDIS - VNP SHOW COMMAND PROCESSING

: DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

: IDENT HISTORY:

: 1.00 14-DEC-79
: DECNET-11M/S V3.0
: DECNET-11M-PLUS V1.0
:
: 3.00 16-APR-82
: DECNET-11M V3.1
: DECNET-11M-PLUS V1.1
:
: 4.00 07-NOV-83
: DECNET-11M V4.0
: DECNET-11M-PLUS V2.0
:
: 5.00 22-JUL-85
: DECnet-11M/S V4.2
: DECnet-11M-Plus V3.0
: DECnet-Micro/RSX V1.0

```

555 .SBTTL $DPRO - SHOW PROCESS
556
557
558
559 $DPRO - SHOW ACTIVE PROCESSES
560 SHOW KNOWN PROCESSES
561 SHOW PROCESS X
562
563 INPUTS: NONE
564
565 OUTPUTS: PROCESS INFORMATION IS DISPLAYED ON USER'S TERMINAL
566
567 ALL REGISTERS ARE DESTROYED
568
569
570
571 $DPRO:: CALL LKTYPE ; SET UP COMMAND TYPE FOR HEADER MESSAGE
572 BCC 5$ ; BR IF SUCCESS
573 ERRPT$ SYNERR,$EROUT ; ELSE SYNTAX ERROR
574 5$: MOV $PDVTA,R4 ; GET ADDRESS OF PDV TABLE OF ADDRESSES
575 MOV $PDVNM,-(SP) ; GET NUMBER OF PROCESSES
576 MOV #TEMP2,R2 ; STORAGE FOR HEADER INTRO MESSAGE
577 TST $QUAL ; SHOW A SPECIFIC PROCESS?
578 BEQ 30$ ; BR IF YES
579 CMP #QF$KNO,$QUAL ; SHOW KNOWN PROCESSES?
580 BEQ 60$ ; BR IF YES
581
582 ; SHOW ACTIVE PROCESSES
583
584 003234 012701 000175' 10$: MOV #PACTIV,R1 ; ELSE MUST BE SHOW ACTIVE PROCESSES
585 003240 112122 MOVB (R1)+,(R2)+ ; STORE THE STRING
586 003242 001376 BNE 10$
587 003244 005001 CLR R1 ; INDICATE HEADER TO BE PRINTED
588 003246 005316 20$: DEC (SP) ; ANY MORE PROCESSES TO CHECK?
589 003250 002455 BLT 100$ ; BR IF NO
590 003252 012402 MOV (R4)+,R2 ; GET ADDRESS OF PDV
591 003254 005762 000000G TST Z.PCB(R2) ; IS THIS PROCESS ON?
592 003260 001772 BEQ 20$ ; BR IF NO
593 003262 CALL DISPRO ; DISPLAY THE INFO FOR THIS PROCESS
594 003266 INC R1 ; INDICATE HEADER NOT TO BE DISPLAYED
595 003270 000766 BR 20$ ; GET NEXT PDV
596
597 ; SHOW SPECIFIC PROCESS
598
599 003272 012701 000236' 30$: MOV #PSPEC,R1 ; GET INTRO STRING FOR HEADER
600 003276 112122 40$: MOVB (R1)+,(R2)+ ; STORE THE STRING
601 003300 001376 BNE 40$
602 003302 012700 000000G 50$: MOV #RQB,R0 ; GET INPUT PARAMETERS
603 003306 005316 DEC (SP) ; ANY MORE PROCESSES?
604 003310 002425 BLT 90$ ; BR IF NO
605 003312 012402 MOV (R4)+,R2 ; GET ADDRESS OF PDV
606 003314 026062 000002 000000G CMP LR.PRO(R0),Z.NAM(R2) ; IS THIS THE ONE TO DISPLAY?
607 003322 001371 BNE 50$ ; BR IF NO
608 003324 005001 CLR R1 ; INDICATE HEADER TO BE DISPLAYED
609 003326 CALL DISPRO ; DISPLAY THE INFO FOR THIS PROCESS
610 003332 000424 BR 100$ ; FINISHED
611

```

.SBTTL DLOEVT - DISPLAY EVENT CHARACTERISTICS

1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131

```

;+
; DLOEVT - DISPLAY EVENT CHARACTERISTICS
; INPUTS:
;   R3 - ADDRESS OF EVENT FILTER BLOCK
; OUTPUTS:
;   THE EVENT MESSAGE IS DISPLAYED
;-
DLOEVT: MOV     F,EVT+0(R3),EVTMSK+0 ; STORE EVENT MASKS
        MOV     F,EVT+2(R3),EVTMSK+2 ; ...
        MOV     F,EVT+4(R3),EVTMSK+4 ; ...
        MOV     F,EVT+6(R3),EVTMSK+6 ; ...
        MOV     #TEMP2,R0 ; POINT TO OUTPUT BUFFER
        BIT     #FF,LIN,F.FLG(R3) ; IS FILTER QUALIFIED BY LINE?
        BEQ     10$ ; BR IF NO
        CALL    BLDLIN ; BUILD LINE ENTITY MESSAGE
        BR      30$ ; CONTINUE
        BIT     #FF,ADD,F.FLG(R3) ; IS FILTER QUALIFIED BY NODE?
        BEQ     20$ ; BR IF NO
        CALL    BLDNOD ; BUILD NODE ENTITY MESSAGE
        BR      30$ ; CONTINUE
        MOV     F,CLS(R3),GCLASS ; ELSE HAVE GLOBAL MASK - SAVE CLASS
        BIC     #FF,MSK,GCLASS ; ISOLATE EVENT CLASS
        MOV     F,EVT+0(R3),GEVMSK+0 ; SAVE GLOBAL FILTER MASKS
        MOV     F,EVT+2(R3),GEVMSK+2 ; ...
        MOV     F,EVT+4(R3),GEVMSK+4 ; ...
        MOV     F,EVT+6(R3),GEVMSK+6 ; ...
        BR      40$ ; CONTINUE
        MOV     F,CLS(R3),R2 ; GET EVENT CLASS
        BIC     #FF,MSK,R2 ; ISOLATE EVENT CLASS
        CMP     R2,GCLASS ; DOES THIS FILTER HAVE A GLOBAL FILTER?
        BNE     40$ ; BR IF NO
        CALL    BLDMSK ; BUILD EVENT MASK
        TST     EVTMSK+0 ; ANY EVENT MASKS SET?
        BNE     60$ ; BR IF YES
        TST     EVTMSK+2 ; ...
        BNE     60$ ; ...
        TST     EVTMSK+4 ; ...
        BNE     60$ ; ...
        TST     EVTMSK+6 ; ...
        BNE     60$ ; ...
        MOV     #LONOEVR1 ; POINT TO 'NO EVENTS' STRING
        MOV     (R1)+,(R0)+ ; STORE STRING IN OUTPUT BUFFER
        BNE     50$ ; BR IF MORE TO STORE
        BR      140$ ; FINISH UP
        CLRB    BITNUM ; START WITH EVENT TYPE 0
        CLRB    FLAGS ; CLEAR FLAGS WORD
        MOV     F,CLS(R3),R1 ; GET EVENT CLASS (BITS 6-15)
        REPT    6 ; ...
        ASR     R1 ; MOVE EVENT CLASS TO BITS 0-9
        ENDR
        CLR     R2 ; SUPPRESS ZEROES

```

VCEX - VNP LOAD CEX ROUTINES
NLBLK - SETUP NTL HOME BLOCK

MACRO V05.03b Saturday 29-Jun-85 02:13 Page 10

N 1

```

510          .SBTTL NLBLK - SETUP NTL HOME BLOCK
511
512          ;+ NLBLK - SETUP NETLDR DATA BLOCK
513
514          : INPUTS:
515          : R1=CETAB ALLOCATION ADDRESS
516          : R2=CETAB ALLOCATION LENGTH
517          : CEXPCB=CEX PCB ADDRESS
518
519          : OUTPUTS:
520          : DATA BLOCK INITIALIZED
521          :-
522 NLBLK:      CLR      $NTLHB+.CXCSR      ; ZERO THE THIRD WORD (NTINIT - $DEUMR)
523 000666     005067 000004G
524
525          .IF DF,R$$MPL
526 CLR      $NTLHB+.CXPCB      ; NO PCB ADDRESS IF RSX-11M-PLUS
527 .IFF
528 000672     016767 000406' 000006G    MOV      CEXPCB,$N' B+.CXPCB ; STORE CEX PCB ADDRESS IN HOME BLOCK
529 .ENDC
530
531 000700     012700 000514'          6$: MOV      #XBLK1,R0      ; GET FIRST VALIDATION BLOCK ADDRESS
532 000704     012703 000010G          MOV      #N$TLHB+.CXSYM,R3 ; ADDRESS OF THIRD WORD OF NTPPOOL
533 000710     016023 000010          10$: MOV      10(R0),(R3)+ ; STORE SYMBOL VALUE
534 000714     011000 000010          MOV      (R0),R0      ; GET NEXT SYMBOL BLOCK ADDRESS
535 000716     001374
536 BNE      10$              ; IF NE, NOT AT END YET
537
538          .IF DF R$$MPL
539 MOV      #VXBLK1,R0      ; GET VECTORED EXEC VALIDATION BLOCK ADDRESS
540 MOV      #N$TLHB+.CXVSM,R3 ; GET NTL BLOCK VECTORED VALIDATION ADDRESS
541 12$: MOV      10(R0),(R3)+ ; STORE SYMBOL
542 MOV      (R0),R0      ; GET NEXT BLOCK
543 BNE      12$
544 .ENDC
545
546 000720     005023
547 000722     022703 000100G          15$: CLR      (R3)+      ; ZERO THE REMAINDER OF THE BLOCK
548 000726     001374          CMP      #N$TLHB+100,R3 ; ...
549 000730     162703 000100          BNE      15$          ; ...
550 000734     010163 000062          SUB      #100,R3      ; BACK TO START OF HOME BLOCK
551 000740     010263 000064          MOV      R1,.CXALL(R3) ; STORE THE CETAB ALLOCATION ADDRESS
552 000744          MOV      R2,.CXLEN(R3) ; AND THE LENGTH OF THE ALLOCATION
                    RETURN

```

VCEX

SYMBOL

SYMBOL

CBLK1

CBLK3

CBLK4

CBLK5

CEPTR

C\$SYM

CEXNA

CEXPC

CF.DD

CF.DY

CF.EI

CF.FR

CF.LO

CF.MD

CF.TI

CLKQB

CL.OP

C.LGT

C.RQ

C.PSL

C.SSH

C.TCE

C.TIM

DSIZE

D\$END

D\$LNA

ERR1

ERR2

ERR3

ERR4

ERR5

ETAB

ETAB2

FE.CE

F.NRE

F.RS

IBLK

IBLK2

IS\$A

LO.CC

LO.CS

LO.C

LO.L

LO.PA

LO.PR

LO.S

LO.S

LO.VI

LR.A

LR.C

LR.C

VCFG -
LOCAL

SYMBOL	VALUE	REFERENCES
CBLK1	000434 R	#6-162 9-424 9-489
CBLK3	000450 R	6-162 #6-165
CBLK4	000464 R	6-165 #6-168
CBLK5	000500 R	6-168 #6-171
CEPTR	000746 R	8-358 #11-564
C-SYM	000466 R	7-292 #9-408
CEXNAM	000376 R	#6-118 9-414 9-441
CEXPCB	000406 R	#6-141 *7-316 10-528
CF.DDM	= 000002	#5-66
CF.DYN	= 000004	#5-66
CF.EIS	= 000010	#5-66
CF.FRK	= 100000	#5-66
CF.LOG	= 000020	#5-66
CF.MDM	= 000001	#5-66
CF.TIM	= 000400	#5-66
CLKQB	000410 R	#6-149 13-578 13-711
CL.OPN	= ***** GX	9-409
CL.LGTH	= 000020	6-149 13-669 13-692 13-719
C.RQT	000002	*13-681
C.PSI	000012	*13-682 *13-683
C.SSHT	= 000004	13-681
C.TCB	000004	*13-684 13-698 13-698
C.TIM	000006	*13-680 *13-686 *13-687 13-700 13-700 13-703 13-703
DSIZE	= ***** GX	14-789
DSEND	= 000104	8-379
D\$LNAM	000006	8-381
ERR1	000034 R	#6-94 14-741
ERR2	000066 R	#6-96 14-753
ERR3	000152 R	#6-98 14-781
ERR4	000224 R	#6-100 14-843
ERR5	000252 R	#6-102 13-721
ETAB1	000716 R	#6-235 9-507
ETAB2	000726 R	#6-243 8-393
FE.CEX	= ***** GX	8-348 8-392
F.NRBD	= ***** GX	*11-580 *11-584 *11-618 *13-711 *13-713 *14-848
F.RSIZ	= ***** GX	*7-318 *7-325 *8-350 *8-363 *8-379 *11-581 *11-585 *12-636 *13-685
		*13-692 *14-786 *14-789 *14-837 11-570
IBLK1	000610 R	#6-206 9-447
IBLK2	000622 R	6-206 #6-209
IS\$AS	= *****	5-65 5-65 7-322
LO.CON	000006	#5-65
LO.CSR	000006	#5-65
LO.CTL	000014	#5-65
LO.LIN	000004	#5-65
LO.PAR	000010	#5-65
LO.PRI	000010	#5-65
LO.SCR	000014	#5-65
LO.STA	000015	#5-65
LO.VCT	000012	#5-65
LR.AST	000026	#5-65
LR.CIR	000002	#5-65
LR.CTL	000004	#5-65

21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50

```

219
220
221
222 ; ERROR MESSAGE FORMAT STRING
223
224 002422 103 157 156 FMT8:: .ASCIZ "Config File -- "
225 002442 103 157 156 FMT8B:: .ASCIZ "Config File -- Illegal "
226 .EVEN
227
228
229
230 ; RECORD BUFFER AND BUFFER LENGTH
231
232 002472 CFGBF:: .BLKB 134.
233 002700 CFGSZ:: .BLKW 1
234
235
236 ; SAVED STACK POINTER FOR ERROR EXIT
237
238 002702 SPSAV: .BLKW 1
239
240
241 ; SYNTAX ERROR FLAG
242
243 002704 SYNERR:: .BLKW 1
244
245 .PSECT

```

```

699
700
701
702
703
704
705
706
707
708
709 001476 012700 002472'
710 001502 016701 002700'
711 001506

      .SBTTL  CFLIN - ECHO LAST RECORD
      ;+
      ; CFLIN - ECHO THE LAST RECORD READ FROM THE CONFIG FILE
      ;
      ; INPUTS:
      ;     NONE
      ;
      ; OUTPUTS:
      ;     NONE
      ;
      ;
      CFLIN:: MOV  #CFGBF,R0      ; GET RECORD BUFFER ADDRESS
              MOV  CFGSZ,R1      ; AND RECORD SIZE
              CALLR $TIOUT      ; PRINT IT ON T1:
  
```

1236
 1237
 1238
 1239
 1240
 1241
 1242
 1243
 1244
 1245
 1246
 1247
 1248
 1249
 1250
 1251
 1252
 1253
 1254
 1255
 1256
 1257
 1258


```

1236 002272 001413      BEQ    10$      ; IF EQ, NO ... TREAT AS PT-TO-PT LINE
1237 002274 016701 000150'  MOV    SCNT,R1      ; GET THE STATION COUNT
1238 002300 005301      DEC    R1        ; REALLY LAST ONE
1239 002302 006301      ASL    R1        ; CONVERT LOGICAL STATION NUMBER TO 2 WORD INDEX
1240 002304 006301      ASL    R1        ;
1241 002306 060001      ADD    RO,R1      ; GET START OF STATION TABLE
1242 002310 062701      ADD    #L.MPF,R1 ;
1243 002314 116761 000000G 000000G  MOVB   .PNUMB,$.COST(R1) ; STORE STATION COST
1244 002322      10$:    RETURN
1245
1246 002324      101$:   MSG$R YP      ; ILLEGAL LINE COST
1247
1248      ;
1249      ; STATION ACTIVE POLLING RATIO (STA$DF)
1250
1251 002332 105767 000001G  ST.APR: TSTB   .PNUMB+1    ; IS IT A BYTE VALUE ?
1252 002336 001007      BNE    101$      ; IF NE, NO ... ERROR
1253 002340 016700 000150'  MOV    SCNT,RO    ; GET CURRENT STATION COUNT
1254 002344 005300      DEC    RO        ; OUR'S IS LAST ONE
1255 002346 116760 000000G 000000G  MOVB   .PNUMB,$$APR(RO) ; SAVE ACTIVE POLLING RATIO
1256 002354      RETURN
1257
1258 002356      101$:   MSG$R YQ      ; ILLEGAL ACTIVE POLLING RATIO

```

ADDDF
 AD.NUL
 \$\$\$CH
 \$\$\$CP
 \$\$\$PR
 \$\$\$TR
 BITS
 BITS1
 BITS2
 BLKSM
 BLKSM
 BRADJ
 CERR
 CFERR
 CFGBF
 CFGEK
 CFGL
 CFGKW
 CFGNA
 CFGST
 CFGSZ
 CFLIN
 CHERR
 CHKBL
 CHKWN
 CHNLM
 CH.DU
 CH.O1
 CH.EC
 CH.MD
 CH.PA
 CH.PR
 CH.PR
 CH.ST
 CH.SY
 CH.TY
 CL.OP
 CNTDF
 COST
 COST1
 CTL
 \$\$\$CK
 \$\$\$OR
 \$\$\$RS
 C.CSR
 C.CTL
 C.PR
 C.VEC
 C1.BS
 C1.DC
 C1.SD
 C1.X2
 DDMDF
 DDM1
 DE.OF
 DE.CN
 DPR

ADDDF	000342R	003	DPR1	000616R	003	FL.LMC=	***** GX	MS.LLC=	***** GX	PF\$FM2=	000200		
AD.NUM	002076R		DTEDES	000140RG	002	FMT8	002422RG	002	MS.MTP=	***** GX	PF\$OFF=	000000	
AS\$CHK=	000300		DTEFLG	000142RG	002	FMT8B	002442RG	002	MS.MUX=	***** GX	PF\$RM1=	000020	
AS\$CPS=	000000		D\$AMXC	000072		FM.8	= 000000		MS.PRI=	***** GX	PF\$RM2=	000040	
AS\$PRI=	000000		D\$AMXH	000074		FM.8B	= 000000		MS.SEC=	***** GX	PF\$SRVR=	000010	
AS\$TRP=	000000		D\$ANN	000000		FNDPDV	004120RG		MS.VCT=	***** GX	PF\$STA=	000004	
BITS	000154R	003	D\$BRPR	000102		F\$SLVL=	000001		MS\$CRB=	000124	PF\$SVC=	040000	
BITS1	000204R	003	D\$BRTM	000100		F.CH1	003734R		MS\$CRX=	000000	PF\$UP	= 000004	
BITS2	000220R	003	D\$DELF	000045		F.CH2	003756R		MS\$FCS=	000000	PF.PSV=	***** GX	
BLKSMN=	000040		D\$DELW	000046		F.FEA	004000R		MS\$MGE=	000000	PF.XDF=	***** GX	
BLKSMX=	002000		D\$END	= 000104		F.MOD	004010R		MS\$NFT=	000000	PLTDF	000356R	
BRADJ	000130R	003	D\$FNB	000034		F.RSIZ=	***** GX		MS\$OVR=	000000	PL.CHN	002130R	
CERR	001234RG		D\$HIOR	000024		F.SET	004020R		NAME	000012R	002	PR\$BED=	000200
CFERR	001244RG		D\$HOST	000022		GETCSR	002562R		NSADJ1	000072		PR\$BEU=	000100
CFGBF	002472RG	002	D\$INAC	000044		G\$STTP=	000000		NSADJ2	000074		PR\$BRD=	000040
CFGEXT	000006R	002	D\$INCT	000042		G\$STSS=	000000		NSCACH	000062		PR\$BRU=	000020
CFGFLG	000136RG	002	D\$IPL	000051		G\$STTK=	000000		NSCRC	000120		PR\$DWN=	000002
CFGKW	000000RG	004	D\$IID	000020		G\$SWRD=	000000		NSHC	000052		PR\$LCC=	000010
CFGNAM	000000R	002	D\$LNAM	000006		HSHADD	000146RG	002	NSHC2	000056		PR\$MOP=	000004
CFGST	000000RG	003	D\$LNUM	000014		HSHSZ	000144RG	002	NSLV1	000002		PR\$UP	= 000001
CFGSZ	002700RG	002	D\$LST	000047		IE.EOF=	***** GX		NSLV2	000010		PS\$CHR=	000016
CFLIN	001476RG		D\$MAXC	000064		IE.RBG=	***** GX		NSMHC1	000036		PS\$FAI=	000014
CHERR	003066R		D\$MAXH	000066		INC	001012R	003	NSMHC2	000044		PS\$NTI=	000006
CHKBLK	004172RG		D\$MAXV	000070		ISSHAR=	000000		NSPLD	000016		PS\$OFF=	000000
CHKWND	004230RG		D\$MLL	000040		ISSRDN=	000000		NSPRI	000076		PS\$STR=	000002
CHNLMX=	007777 G		D\$MNOD	000041		K\$SCNT=	177546		NSROA1	000022		PS\$UP	= 000012
CH.DUP=	000001		D\$NA	000062		K\$SCSR=	177546		NSROA2	000030		PS\$VER=	000010
CH.D11=	000100		D\$NBEA	000056		K\$SLDC=	000000		NSRTMX	000014		PS\$WT	= 000004
CH.ECH=	000002		D\$NBRA	000054		K\$STPS=	000074		NSRTM1	000015		PT\$BRD=	000200
CH.MDT=	000020		D\$NEND=	000054		LCTIM	001044R	003	NSRTM2	000015		PT\$DRT=	000100
CH.PAR=	000100		D\$NLN	000030		LD\$LP	= 000000		NSRT1	000000		PT\$END=	000004
CH.PRM=	000200		D\$NN	000060		LLCDF	000724R	003	NSRT2	000006		PT\$SLV1=	000002
CH.PRT=	000007		D\$OUTT	000043		LLCDF1	000766R	003	NSCTL	000112		PT\$SLV2=	000001
CH.STA=	000040		D\$RETF	000050		LNKEND	004030RG		NSLTC	000100		PT\$PH3=	000010
CH.SYN=	000004		D\$RNN	000002		L\$ASG=	000000		NSTNC	000114		PT\$XAR=	000020
CH.TYP=	000010		D\$RTMR	000076		L\$SDRV=	000000		NSTRC	000106		PX\$BLK=	000040
CL.OPN=	***** GX		D\$SEG	000036		L\$SP11=	000001		NSTTGB	000110		PX\$DLM=	000200
CNTDF	000456R	003	D\$SER	000032		L\$S11R=	000000		NSVER	000066		PX\$SVC=	000100
COST	000566R	003	D\$SQRL	000052		L.CH1	003714R		NSXLEN	000124		PX\$CHN	000004
COST1	000576R	003	D\$SUBG=	177514		L.CH2	003724R		NS\$ACC=	000001		PS\$CNT	000005
CTL	000015R	002	D\$ISK=	000000		L.COST=	***** GX		NS\$BUF=	000001		PS\$CTR	000034
C\$CKP=	000000		D\$S11=	000001		L.CTL	= ***** GX		NS\$LDV=	000001		PS\$DRT	000015
C\$ORE=	000400		D\$SYNC=	000000		L.DDM	= ***** GX		NS\$MCP=	000001		PS\$DRT	000036
C\$RSH=	177564		D\$SYNM=	000000		L.DLC	= ***** GX		NS\$MLL=	000001		PS\$FLG	000012
C.CSR	002522R		D.EXT	001624R		L.MPF	= ***** GX		NS\$MOV=	000010		PS\$WD	000030
C.CTL	001706R		D.NAME	001512R		L.MPE	001512R		NS\$NCT=	000001		PS\$CCB	000046
C.PRI	002626R		END	000652R	003	L.NSTA=	***** GX		NS\$PEM=	000001		PS\$IPL	000014
C.VECT	002432R		ERRBUF	000027R	002	L.UNT	= ***** GX		OPEN	000730R	003	PSLCD	000002
C1.BSY=	000002		ERRMSG	000017R	002	MAXCST=	000031		PECH	000626R	003	PSLEN	000052
C1.DCP=	000001		EXT	001022R	003	MAXID	= 000020		PECH1	000636R	003	PSLST	000000
C1.SDL=	000003		E\$XPR=	000000		MAXNDC=	000020		PF\$BLK=	020000		PSNRN1	000040
C1.X25=	000004		FEAD	000662R	003	MAXWND=	000007		PF\$CLC=	010000		PSOCCB	000050
DDMDF	000372R	003	FG.X2P=	000001 G		MOVE	001436RG		PF\$DLM=	100000		PSPFQ	000006
DDM1	000410R	003	FG.X3P=	000002 G		MS.CIR=	***** GX		PF\$EIP=	000002		PSPKSZ	000044
DE.OFF=	000002 G		FL.FDX=	***** GX		MS.CSR=	***** GX		PF\$FNB=	000001		PSRMX1	000016
DE.ON	= 000001 G		FL.HDX=	***** GX		MS.EXT=	***** GX		PF\$FAI=	004000		PSRMX2	000020
DPR	000606R	003	FL.KMX=	***** GX		MS.INC=	***** GX		PF\$FM1=	000100		PSRPR1	000042

.TITLE VCFP - VNP PSI CONFIGURATION FILE SCAN
 .IDENT /V05.00/
 .ENABL LC

;[TD001]
 ;**-1

COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
 DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
 ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
 INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
 COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
 OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
 TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
 AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
 CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
 SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

MODULE DESCRIPTION:

VNP - CONFIG FILE SCANNER FOR "SET SYSTEM" FOR PSI COMPONENTS

DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

IDENT HISTORY:

- 1.00 01-MAR-82
 VERSION 3.1 RELEASE
- 3.00 16-APR-82
 DECNET-11M V3.1
 DECNET-11M-PLUS V1.1
- 3.01 11-OCT-82
 Changes for X25 Gateway
 [Tim Dixon FRSJ]
- 3.02 5-APR-82
 CORRECT GCL LEVEL BUG
- 4.00 07-NOV-83
 DECNET-11M V4.0
 DECNET-11M-PLUS V2.0
- 5.00 22-JUL-85
 DECnet-11M/S V4.2
 DECnet-11M-Plus V3.0
 DECnet-Micro/RSX V1.0

;[TD001]
 ;[TD001]
 ;[TD001]
 ;[TD001]
 ;**-1

```

503                               .SBTTL   CFLIN - ECHO LAST RECORD
504                               ;+
505                                :CFLIN - ECHO THE LAST RECORD READ FROM THE CONFIG FILE
506                                :
507                                :INPUTS:
508                                    NONE
509                                :
510                                :OUTPUTS:
511                                    NONE
512                                :-
513      000632          FNLIN:: CALL    $FNOUT           ; PRINT FILE NAME
514     012700  000130' CFLIN:: MOV    #CFGBF,R0        ; GET RECORD BUFFER ADDRESS
515     016701  000336'              MOV    CFGSZ,R1       AND RECORD SIZE
516            000646             CALLR   $TIOUT         ; PRINT IT ON TI:

```

VCFF
SYME
SYME
BLKS
BLKS
BLKS
BLKS
CERR
CFER
CFGE
CFGE
CFGE
CFGE
CFLE
CHKR
CHKW
CHNI
CL
COUN
CTIN
CUNN
CVTE
DARL
DARL
DEVA
DEVA
DEVA
DE.
DSA
DSA
DSA
DSA
DSA
DSA
DSA
DSC
DSC
DSC
DSF
DST
DST
DST
DST
DST
DST
DST
DS.

VCFP CREATED BY MACRO ON 29-JUN-85 AT 02:18

PAGE 1

SYMBOL CROSS REFERENCE

CREF 04.00

SYMBOL	VALUE	REFERENCES
BLKSMN	= 000040	#6-89 14-555
BLKSMX	= 002000	#6-90 14-557
BLKSZD	000360 RG	#7-156
BLKSZM	000362 RG	#7-157
CERR	000534 RG	8-259 8-260 8-261 8-262 8-263 8-264 8-265 8-266 8-267
		8-268 8-269 8-270 8-271 8-272 8-276 #12-476
CFERR	000544 RG	8-273 8-274 8-275 #12-479
CFGBF	000130 RG	#7-131 11-444 13-514
CFGEXT	000006 R	#7-116 10-421
CFGNAM	000000 R	#7-115 10-420
CFGSZ	000336 RG	#7-132 *11-448
CFLIN	000636 RG	8-259 8-260 8-261 13-515 8-262 8-263 8-264 8-265 8-266 8-267
		8-276 #13-514
CHKBLK	000742 RG	#14-552
CHKWND	001000 RG	#14-566
CHNLMX	= 007777 G	#6-91
CL.OPN	= ***** GX	9-305
COUNT	000356 RG	#7-150 15-594 *18-695
CTIM	000400 RG	#7-170
CUNMMX	= 000006 G	#6-92
CVTBUF	000357 RG	#7-151 *18-687 18-691
DARDEN	= 000516 RG	#7-220
DARMLN	= 000017 G	#7-217 7-218 7-224
DEVCTL	000700 RG	#14-533
DEVNAM	000652 RG	#14-524
DEVUNT	000716 RG	#14-540
DE.OFF	= 000002 G	#7-166
DE.ON	= 000001 G	#7-165
DSACGL	000533 RG	7-203 #7-225
DSACUG	000534 RG	#7-226
DSAHCT	000521 RG	#7-222
DSALCT	000520 RG	#7-221
DSARCT	000522 RG	7-212 #7-223
DSARDP	000523 RG	#7-224
DSARDT	000506 RG	#7-218
DSASHI	000556 RG	#7-229
DSASLO	000554 RG	#7-228
DSCMCT	000560 RG	7-210 #7-233
DSCMSK	000561 RG	#7-234
DSCVAL	000622 RG	#7-236
DSCVCT	000621 RG	7-211 #7-235
DSFLG	000404 RG	#7-180
DSTEND	= ***** GX	9-387
DSTNAM	000430 RG	#7-186
DSTNML	000427 RG	#7-185 7-202
DSTOBJ	000451 RG	#7-188
DSTPRI	000450 RG	#7-187
DSTTKL	000406 RG	#7-184 7-204
DSTTSK	000407 RG	#7-184
DSTVR0	000452 RG	#7-201
DSTVR1	000470 RG	#7-209
DS.X25	= 000001 G	#7-181

VCIO - LOAD CETAB IMAGE MACRO V05.03b Saturday 29-Jun-85 02:18 Page 10

```

265      ;+
266      $DECHM - ALLOCATE DECNET HOME BLOCK FROM DSR
267
268      INPUTS -
269              NONE
270
271      OUTPUT -
272              DECNET HOME BLOCK ALLOCATED
273              $DECTPT POINTS TO HOME BLOCK
274              RO,R1 DESTROYED
275      ; -
276
277
278
279 000416 012701 000104 $DECHM: MOV    #D$END,R1      ; GET NUMBER OF BYTES TO ALLOCATE
280 000422          CALL  $ALL15    ; ALLOCATE FROM DSR (NOTE FE.CEX NOT SET)
281 000426          BCC   10$       ; IF SUCCESS - BRANCH
282
283 000430 012700 000044      MOV    #ERR36,R0      ; ALLOCATION FAILURE
284 000434          RETURN
285
286 000436 010067 000000G    10$: MOV    R0,$DECTPT      ; SET UP POINTER
287 000442 012700 000000G      MOV    #B$FR,R0      ; GET I/O BUFFER ADDRESS
288 000446 105020          CLR    (R0)+      ; INIT BLOCK
289 000450 005301          DEC    R1          ; COUNT ONE BYTE
290 000452 001375          BNE    15$         ; IF MORE - BRANCH
291 000454 012700 000000G      MOV    #B$FR,R0      ; GET STATING ADDRESS
292 000460 016701 000000G      MOV    $DECTPT,R1    ; ...
293
294 000464 062701 000004      ADD    #D$RNN+2,R1    ; SET UP REMOTE LIST POINTERS
295 000470 010160 000002      MOV    R1,D$RNN(R0)    ; ...
296
297 000474 112760 000252 000024      MOVB  #252,D$HIOR(R0) ; SET UP ETHERNET ADDRESS HIGH ORDER BYTES
298 000502 105060 000025          CLR    D$HIOR+1(R0) ; ...
299 000506 112760 000004 000026      MOVB  #4,D$HIOR+2(R0) ; ...
300 000514 105060 000027          CLR    D$HIOR+3(R0) ; ...
301
302 000520          PUTRC  $DECTPT,#D$END      ; WRITE OUT DECNET HOME BLOCK TO IMAGE
303
304 000540          RETURN
305
306
307          .END

```

VCLQF
Symbo

AUXPDR
ASSCH
ASSCH
ASSPDR
CLQRCH
ASSHT
C55RCH
C55OR
C55SR
C.A.R
C.A.S
C.A.S
C.D.S
C.E.F
C.L.G
C.L.N
C.M.R
C.R.Q
C.R.S
C.S.R
C.S.S
C.S.S
C.S.Y
C.S.Y
C.C.I
C.V.I
D55B
D55I
D55L
D55Y
D55Y
F55X
F55X
F55L
F.R.S
G55T
G55T
G55T
G55W
I55R
I55R
K55C
K55C
K55L
K57R
LD5L
LD.A
LD.C
LD.C
LD.L
LD.L
LD.M
LD.N
LD.N

1. AB

VCLQRM - REMOVE ENTRIES FROM CL MACRO V05.03b Saturday 29-Jun-85 02:18 Page 8-1
Symbol table

AUXPDV= 000000R	LD.OBJ= 000032	LR.NAM= 000012	N\$NCT= 000001	OP\$SEG= 000400
A\$CHK= 000000	LD.PRO= 000026	LR.PAS= 000041	N\$PEM= 000001	OP\$SER= 000400
A\$CPS= 000000	LO.ADD= 000022	LR.PRO= 000002	OF\$INS= 000001	OP\$SIN= 000010
A\$PRI= 000000	LO.ADR= 000052	LR.STS= 000000	OF\$LOG= 000100	OP\$STA= 000004
A\$TRP= 000000	LO.CLS= 000002	LR.TRI= 000006	OF\$OFF= 000002	OP\$TPA= 000020
CLQREM= 000002RG	LO.CDN= 000006	LR.TYP= 000002	OF\$ON= 000000	OP\$TRI= 000040
C\$CKP= 000000	LO.COP= 000010	LR.UCB= 000016	OF\$SMC= 000200	OP\$UCS= 000020
C\$ORE= 000400	LO.COS= 000020	LR.UIC= 000020	OP\$ADD= 000200	OP\$USE= 000002
C\$RSH= 177564	LO.CSR= 000010	LR.UNT= 000005	OP\$ALL= 000200	OP\$VEC= 000040
C.ARS= 000014	LO.CTL= 000032	LS.ALI= 000016	OP\$CCS= 000004	OP\$VER= 000040
C.AST= 000012	LO.DES= 000002	LS.CEX= 000001	OP\$CIE= 004000	OP\$XAC= 000001
C.CSTP= 000012	LO.DTE= 000047	LS.CIR= 000033	OP\$CON= 000400	OP\$XPR= 000002
C.DST= 000016	LO.EVT= 000004	LS.CX0= 010000	OP\$COP= 000001	OP\$XPS= 000004
C.EFN= 000003	LO.GRO= 000024	LS.CX5= 000400	OP\$COS= 000100	OP\$X9S= 000010
C.LGTH= 000020	LO.HOS= 000040	LS.ECH= 001000	OP\$DST= 000200	P\$P45= 000000
C.LNK= 000000	LO.HTM= 000032	LS.FDX= 004000	OP\$DTE= 000040	P\$WRD= 000000
C.MRK= 000000	LO.INA= 000062	LS.HDX= 010000	OP\$DUP= 000002	Q\$ACT= 000001
C.RQT= 000002	LO.INC= 000060	LS.HLP= 000012	OP\$EST= 000100	Q\$ALL= 000002
C.RSI= 000012	LO.LDT= 000046	LS.LIN= 000003	OP\$EVE= 010000	Q\$KNO= 000004
C.SCHD= 000002	LO.LID= 000030	LS.LMC= 000007	OP\$FIL= 000200	Q\$LOO= 000010
C.SRC= 000014	LO.LIN= 000004	LS.LOG= 000020	OP\$GRP= 000100	Q\$OPT= 000010
C.SSHT= 000004	LO.LTM= 000034	LS.MOD= 000036	OP\$HOS= 000001	RQB= ***** GX
C.SUB= 000012	LO.MAC= 000022	LS.NOD= 000014	OP\$HTM= 000002	R\$DER= 000000
C.SYST= 000006	LO.MDE= 000024	LS.NTI= 000013	OP\$INA= 004000	R\$K11= 000001
C.SYTK= 000010	LO.NAM= 000044	LS.OBJ= 000022	OP\$INC= 001000	R\$SND= 000000
C.CB= 000004	LO.NET= 000002	LS.OPT= 000400	OP\$KDS= 002000	R\$11M= 000000
C.IIM= 000016	LO.NNM= 000022	LS.PRO= 000002	OP\$KDT= 000400	SYSFDB= ***** GX
C.UIC= 000016	LO.OFL= 000012	LS.TOP= 002000	OP\$KEV= 001000	S\$WRG= 000000
D\$BUG= 177514	LO.ONM= 000004	LS.TST= 000010	OP\$KGR= 001000	S\$YSZ= 007600
D\$ISK= 000000	LO.OUT= 000056	LS.UNF= 020000	OP\$KNT= 010000	TY\$CHA= 000400
D\$L11= 000001	LO.OWN= 000026	LS.XIT= 000011	OP\$KSK= 002000	TY\$EVE= 001000
D\$YNC= 000000	LO.PAR= 000010	LX.ALI= 000017	OP\$LCT= 000200	TY\$STA= 002000
D\$YNM= 000000	LO.PRI= 000012	LX.CEX= 000004	OP\$LNE= 000040	TY\$SUM= 000000
E\$XPR= 000000	LO.RET= 000064	LX.CIR= 000034	OP\$LOC= 000001	T\$KMG= 000000
FNDPCV= 000242R	LO.RPA= 000030	LX.LIN= 000006	OP\$LST= 000002	T\$MIN= 000000
F\$SLVL= 000001	LO.SCR= 000016	LX.LOG= 000021	OP\$LTM= 000001	V\$CTR= 001000
F.RSIZ= ***** GX	LO.SEG= 000054	LX.NCJ= 000015	OP\$LVL= 000010	WC.CNT= 000400
G\$TPP= 000000	LO.SKA= 000014	LX.OBJ= 000023	OP\$MAC= 000010	WC.EVE= 004000
G\$TSS= 000000	LO.SKN= 000014	LX.PRO= 000005	OP\$MCO= 000002	WC.TRI= 002000
G\$TTK= 000000	LO.SNM= 000035	L\$ASG= 000000	OP\$MDE= 000400	WC.UNT= 001000
G\$WRD= 000000	LO.TPA= 000020	L\$DRV= 000000	OP\$MLI= 000004	X\$DBT= 000000
I\$RAR= 000000	LO.TRB= 000034	L\$P11= 000001	OP\$MON= 000100	Z.DSP= ***** GX
I\$RDN= 000000	LO.TRI= 000030	L\$11R= 000000	OP\$MST= 000020	Z.NAM= ***** GX
K\$CNT= 177546	LO.UNT= 000033	M\$CRB= 000124	OP\$NAM= 000004	\$BFR= ***** GX
K\$CSR= 177546	LO.VCT= 000014	M\$CRX= 000000	OP\$NET= 004000	\$CEXND= ***** GX
K\$LDC= 000000	LO.VER= 000042	M\$FCS= 000000	OP\$NLI= 000002	\$DEA16= ***** GX
K\$TPS= 000074	LR.ACC= 000020	M\$MGE= 000000	OP\$NOD= 000020	\$GETRV= ***** GX
LD\$LP= 000000	LR.ACO= 000052	M\$NET= 000000	OP\$OUT= 002000	\$PDVNM= ***** GX
LD.ALI= 000030	LR.ADD= 000012	M\$OVR= 000000	OP\$OWN= 000020	\$PDVTA= ***** GX
LD.CEX= 000024	LR.ALI= 000002	N\$ACC= 000001	OP\$PAR= 000010	\$PUTRC= ***** GX
LD.CIR= 000035	LR.BLK= 000010	N\$BUF= 000001	OP\$PRI= 000010	\$RADDR= ***** GX
LD.LIN= 000025	LR.CTL= 000004	N\$LDV= 000001	OP\$RET= 010000	.CLKHD= ***** GX
LD.LOG= 000031	LR.DES= 000010	N\$MCF= 000001	OP\$RPA= 000010	.MGMGE= ***** GX
LD.MOD= 000037	LR.HLP= 000002	N\$MLL= 000001	OP\$SCO= 000001	.PDVTA= ***** GX
LD.NOD= 000027	LR.LIN= 000002	N\$MOV= 000010		

. ABS. 000106 000 (RW,I,GBL,ABS,OVR)

```

144 .SBTTL ERROR MESSAGES
145 ;
146 ; ERROR MESSAGES
147 ;
148 002204 NTLERS $ 2A,4A,DVERR,REP4A,,<undefined>
149 002226 NTLERS $ 2B,4,DVERR,$PRV.N,<Open failure (-***.)>
150 002264 NTLERS $ 2C,4,DVERR,$PRV.N,<Read failure (-***.)>
151 002322 NTLERS $ 2D,4,DVERR,$REP.P,<Read overflow>
152 002350 NTLERS $ 2E,4,DVERR,$REP.P,<Format error>
153 002376 NTLERS $ 2F,6,DVERR,$PRV.N,<Open failure (-***.)>
154 002434 NTLERS $ 2H,6,DVERR,$REP.C,$CLDEQ,<not contiguous>
155 002464 NTLERS $ 2J,6,DVERR,$CUR.N,$CLDEQ,<Label block read failure (-***.)>
156 002536 NTLERS $ 2K,6,DVERR,$REP.C,$CLDEQ,<not a valid binary image.>
157 002600 NTLERS $ 2L,9,DVERR,REP9,$CLDEQ,<allocation failure>
158 002634 NTLERS $ 2M,6,DVERR,$CUR.N,DVDEA,<Read failure (-***.)>
159 002672 NTLERS $ 2N,4A,DVERR,REP4A,,<odd>
160 002706 NTLERS $ 2P,4A,DVERR,REP4A,,<outside table limits.>
161 002744 NTLERS $ 2Q,9,DVERR,REP9,DVDEA,<illegal unit count.>
162 003000 NTLERS $ 2R,9,DVERR,REP9,DVDEA,<D.UCB field outside table limits.>
163 003052 NTLERS $ 2S,9,DVERR,REP9,DVDEA,<UCB address odd.>
164 003104 NTLERS $ 2T,9,DVERR,REP9,DVDEA,<UCB address outside table limits.>
165 003156 NTLERS $ 2U,9,DVERR,REP9,DVDEA,<U.SCB field outside table limits.>
166 003230 NTLERS $ 2V,9,DVERR,REP9,DVDEA,<SCB address odd.>
167 003262 NTLERS $ 2W,9,DVERR,REP9,DVDEA,<SCB address outside table limits.>
168 003334 NTLERS $ 2X,9,DVERR,REP9,,<driver already resident.>
169 003376 NTLERS $ 2Y,4A,DVERR,REP4A1,,<undefined>
170 ;
171 ; ERROR MESSAGE FORMAT STRINGS
172 ;
173 003420 052 040 123 FMT4A: .ASCIIZ '* Symbol table file -- symbol * '
174 003461 052 040 123 FMT4: .ASCIIZ '* Symbol table file --
175 003511 052 040 111 FMT6: .ASCIIZ '* Image file -- '
176 003532 052 072 040 FMT9: .ASCIIZ '*: Device tables -- '
177 .EVFN
178 ;
179 000000 .PSECT ..BUF,D
180 ;
181 ; DISK BUFFER FOR LABEL BLOCK READ
182 ;
183 ;
184 000000 DSKBUF: .BLKW 1
185 ;
186 000000 .PSECT
  
```


VDEV - VNP LOAD DEVICE STRUCTUR MACRO V05.03b Tuesday 03-Sep-85 10:28 Page 15
 DVDEA - DEALLOCATE DATA STRUCTURES

```

674      .SBTTL  DVDEA - DEALLOCATE DATA STRUCTURES
675      ;+
676      ; DVDEA - DEALLOCATE DATA STRUCTURES AND CLOSE IMAGE FILE
677      ;
678      ; INPUTS:
679      ; $LBUF=BUFFER ADDRESS
680      ; $LLEN=BUFFER LENGTH
681      ;
682      ; OUTPUTS:
683      ; NONE
684      ; -
685      DVDEA:
686      MOV     $LBUF,R0      ; GET BUFFER ADDRESS
687      MOV     $LLEN,R1      ; AND LENGTH
688      SWTCHI  $BFR          ; DEFAULT TO $BFR
689      CALL    $DEA16        ; DE-ALLOCATE DEVICE DATA STRUCTURES
690      SWTCHI  #DEVBUF        ; RETURN TO LOCAL BUFFER
691      CALL    $CLDEQ         ; DEQUEUE GCL FILE
692      RETURN
693
685 002142
686 002142 016700 000000G
687 002146 016701 000000G
688 002152
689 002160
690 002166
691 002172
692 002200
693 002206
694 002212

```

```

54 .SBTTL MACRO CALLS
55
56 .MCALL ERMSG$,ERBLK$,ERRPT$,VNPDF$,GETAD$,GETRV$,SWTCH1$,SAVRG$,RESRG
57 .MCALL OBJDF$,ANBDF$,NSFDF$,RNBDF$,SLTDF$,FLTDF$,ASRS$
58 .MCALL DDCDF$,PCLDF$,UCBDF$,DCBDF$,TCBDF$,DIR$,QLOW$,CEACCS$
59 .MCALL ASL$,CTRDF$,DMPDF$,PHBDF$,PLIDF$,DTEF$,CUGDF$,NWDF$,DSTDF$
60 .MCALL RDTDF$,X29DF$,PVCDF$,ECDDB$,XPDDB$,PLBDF$,DHBDF$
61
62 .IF NDF R$$MPL ;[MP01]
63 .MCALL NKRDF$ ;[MP01]
64 000000 ; DEFINE NETWORK KRB OFFSETS ;[MP01]
65
66 .IFF ;[MP01]
67 .MCALL KRBDF$ ;[MP01]
68 KRBDF$ ;[MP01]
69 .ENDC ;[MP01]
70
71
72 .MCALL RNWDF$ ;[TD001]
73 .MCALL VNPDBG ;[TD001]
74
75 000000 ANBDF$ ; ALIAS NAME TABLE SYMBOL DEFINITIONS
76 000000 CTRDF$ ; COUNTER BLOCK SYMBOL DEFINITIONS
77 000000 CUGDF$ ; CLOSED USER GROUP SYMBOL DEFINITIONS
78 000000 DCBDF$ ; DCB SYMBOL DEFINITIONS
79 000000 DDCDF$ ; DDCMP SYMBOL DEFINITIONS
80 000000 DMPDF$ ; DMP SYMBOL DEFINITIONS
81 000000 DSTDF$ ; DESTINATION DESCRIPTOR DEFINITIONS
82 000000 DTEF$ ; DTE SYMBOL DEFINITIONS
83 000000 ECDDB$ ; ECL DATA BASE SYMBOL DEFS
84 000000 FLTDF$ ; FILTER CONTROL BLOCK DEFINITIONS
85 000000 NSFDF$ ; FEATURE WORD DEFINITIONS FOR NSP
86 000000 NWDF$ ; NW LINE TABLE SYMBOL DEFINITIONS
87 000000 OBJDF$ ; OBJECT TABLE SYMBOL DEFINITIONS
88 000000 PCLDF$ ; PCL SYMBOL DEFINITIONS
89 000000 PHBDF$ ; PSI HOME BLOCK SYMBOL DEFINITIONS
90 000000 PLBDF$ ; DEFINE PLB OFFSETS
91 000000 PLIDF$ ; PLI'S DDB SYMBOL DEFINITIONS
92 000000 PVCDF$ ; PVC NAME BLOCK SYMBOL DEFINITIONS
93 000000 RDTDF$ ; RDT SYMBOL DEFINITIONS
94 000000 RNBDF$ ; TE NAME BLOCK SYMBOL DEFINITIONS
95 000000 RNWDF$ ; Remote network block symbol definition:[TD001]
96 000000 SLTDF$ ; SYSTEM LINE TABLE SYMBOL DEFINITIONS
97 000000 TCBDF$ ; TCB SYMBOL DEFINITIONS
98 000000 UCBDF$ ; UCB SYMBOL DEFINITIONS
99 000000 VNPDF$ ; VNP SYMBOL DEFINITIONS
100 000000 XPDDB$ ; XPT DATA BASE SYMBOL DEFINITIONS
101 000000 X29DF$ ; X29 SYMBOL DEFINITIONS
102 000000 DHBDF$ ; DECNET HOME BLOCK DEFINITIONS
103

```

VDIS - VNP SHOW COMMANDS
\$DPRO - SHOW PROCESS

MACRO V05.03b Monday 15-Jul-85 12:14 Page 11-1

N 15

```
612 ; SHOW KNOWN PROCESSES
613
614 003334 012701 000216' 60$: MOV #PKOWN,R1 ; GET STRING FOR INTRO IN HEADER MESSAGE
615 003340 112122 70$: MOV (R1)+,(R2)+ ; STORE THE STRING
616 003342 001376 ;
617 003344 005001 ;
618 003346 005316 80$: DEC (SP) ; INDICATE HEADER TO BE DISPLAYED
619 003350 002415 ; ANY MORE PROCESSES?
620 003352 012402 ; BR IF NO
621 003354 ; GET ADDRESS OF PDV
622 003360 005201 ; DISPLAY THE PROCESS INFORMATION
623 003362 000771 ; INDICATE HEADER NOT TO BE DISPLAYED
624 ; GET NEXT PDV
625
626 ; PROCESS NOT FOUND
627
628 003364 90$: DIR$ #TIDET ; DETACH TERMINAL
629 003372 005726 TST (SP)+ ; CLEAN UP STACK
630 ERRPT$ PERR1,$EROUT ; DISPLAY ERROR MESSAGE
631
632 003404 100$: DIR$ #TIDET ; DETACH TERMINAL
633 003412 005726 TST (SP)+ ; CLEAN UP STACK
634 RETURN ; FINISHED
635
636 .ENABL LC
637
638 ; ERROR MESSAGES
639
640 003416 ERMSG$ <Process not in system>
641 003443 ERBLK$ PERR1
642
643 .DSABL LC
```


1132	006154				CALL	\$CBDMG	:	CONVERT TO ASCII
1133	006160	720	000056		MOVB	#PERIOD,(R0)+	:	STORE PERIOD AFTER EVENT CLASS
1134	006164	3727	173014	000100 70\$:	CMPB	BITNUM,#MSKLEN	:	ALL BITS PROCESSED?
1135	006172	103065			BHIS	130\$:	BR IF YES
1136	006174				CALL	CHKBIT	:	TEST BIT TO SEE IF IT IS ON
1137	006200	001455			BEQ	110\$:	BR IF NOT ON
1138	006202	132767	000001	173364	BITB	#F.STRT,FLAGS	:	HAS FORMATTING BEGUN ALREADY?
1139	006210	001016			BNE	80\$:	BR IF YES
1140	006212	116701	172766		MOVB	BITNUM,R1	:	GET EVENT TYPE NUMBER
1141	006216	005002			CLR	R2	:	ZERO SUPPRESSION
1142	006220				CALL	\$CBDMG	:	CONVERT TO ASCII
1143	006224	152767	000001	173342	BISB	#F.STRT,FLAGS	:	INDICATE FORMATTING HAS BEGUN
1144	006232	152767	000002	173334	BISB	#F.PREV,FLAGS	:	INDICATE THIS EVENT TYPE IS SET
1145	006240	105067	172741		CLRB	PH.VNT	:	INDICATE FIRST NUMBER IN RANGE
1146	006244	000435			BR	120\$:	CONTINUE
1147								
1148	006246	132767	000002	173320 80\$:	BITB	#F.PREV,FLAGS	:	WAS PREVIOUS NUMBER SET?
1149	006254	001015			BNE	90\$:	BR IF YES
1150	006256	112720	000054		MOVB	#COMMA,(R0)+	:	SEPARATE WITH A COMMA
1151	006262	116701	172716		MOVB	BITNUM,R1	:	GET EVENT TYPE NUMBER TO DISPLAY
1152	006266	105067	172713		CLRB	PREVNT	:	INDICATE VALUE ALREADY DISPLAYED
1153	006272	005002			CLR	R2	:	ZERO SUPPRESSION
1154	006274				CALL	\$CBDMG	:	CONVERT TO ASCII
1155	006300	152767	000002	173266	BISB	#F.PREV,FLAGS	:	INDICATE THIS EVENT TYPE IS SET
1156	006306	000414			BR	120\$:	CONTINUE
1157								
1158	006310	126027	177777	000055 90\$:	CMPB	-1(R0),#DASH	:	IS DASH ALREADY INSERTED?
1159	006316	001402			BEQ	100\$:	BR IF YES
1160	006320	112720	000055		MOVB	#DASH,(R0)+	:	STORE DASH
1161	006324	116767	172654	172653 100\$:	MOVB	BITNUM,PREVNT	:	SAVE EVENT TYPE NUMBER
1162	006332	000402			BR	120\$:	CONTINUE
1163								
1164	006334			110\$:	CALL	CKPREV	:	CHECK FOR DISPLAY OF PREVIOUS EVENT TYPE
1165	006340	105267	172640	120\$:	BITNUM		:	UPDATE BIT NUMBER
1166	006344	000707			BR	70\$:	
1167								
1168	006346			130\$:	CALL	CKPREV	:	SEE IF LAST EVENT TYPE MUST BE DISPLAYED
1169	006352	105020			CLRB	(R0)+	:	CREATE ASCII STRING
1170	006354			140\$:	ERRPT\$	LOMSG4,MSGOUT	:	DISPLAY EVENT MESSAGE
1171	006364				RETURN			

```

1701
1702
1703          ; DISPLAY MAXIMUM HOPS, MAXIMUM VISITS
1704          ;
1705
1706 011166      GETRV  $DECPT,#D$END      ; READ DECNET HOME BLOCK
1707 011206      MOV   $BFR+D$MAXH,TEMP4  ; STORE MAX HOPS FOR DISPLAY
1708 011214      MOV   $BFR+D$MAXV,TEMP3  ; STORE MAX VISITS
1709
1710 011222      ERRPT$ NMSG7,MSGOUT      ; DISPLAY MESSAGE
1711 011232      BR    404$
1712
1713 011234      GETRV  $DECPT,#D$END      ; READ DECNET HOME BLOCK IF F...
1714 011254      MOV   $BFR+D$SEG,TEMP2   ; COPY SEGMENT SIZE
1715 011262      BNE   405$                ; IF NOT DEFAULT - BRANCH
1716 011264      MOV   $RDBSZ,TEMP2       ; CALCULATE DEFAULT
1717 011272      SUB   #18,TEMP2
1718 011300      405$: CLR   TEMP3
1719 011304      BISB  $BFR+D$RETF,TEMP3   ; INIT LOCATION
1720 011312      CLR   TEMP4              ; COPY RETRANSMIT FACTOR
1721 011316      BISB  $BFR+D$INCT,TEMP4   ; INIT LOCATION
1722 011324      CLR   TEMP5              ; COPY INCOMING TIMER
1723 011330      BISB  $BFR+D$OUTT,TEMP5   ; INIT LOCATION
1724 011336      CLR   TEMP6              ; COPY OUTGOING TIMER
1725 011342      BISB  $BFR+D$INAC,TEMP6   ; INIT LOCATION
1726 011350      CLR   R1                  ; COPY INACTIVITY TIMER
1727 011352      BISB  $BFR+D$DELF,R1      ; INIT REG
1728 011356      MOV   #1,R0              ; COPY DELAY FACTOR
1729 011362      ADD   #4,R1               ; CALCULATE ACTUAL VALUE
1730 011366      406$: ASL   R0
1731 011370      DEC   R1
1732 011372      BGT   406$
1733 011374      CLR   TEMP7              ; INIT LOCATION
1734 011400      MOV   R0,TEMP7            ; COPY ACTUAL DELAY FACTOR VALUE
1735 011404      ERRPT$ NMSG13,MSGOUT      ; DISPLAY EXECUTOR INFO
1736 011414      MOV   #1,R0              ; SET UP REG
1737 011420      CLR   R1                  ; INIT REG
1738 011422      BISB  $BFR+D$DELW,R1      ; COPY DELAY WEIGHT
1739 011426      BEQ   408$                ; IF ZERO - BRANCH
1740 011430      407$: ASL   R0
1741 011432      DEC   R1
1742 011434      BGT   407$
1743 011436      CLR   TEMP2              ; INIT LOCATION
1744 011442      MOV   R0,TEMP2            ; COPY ACTUAL DELAY WEIGHT
1745 011446      CLR   TEMP3              ; INIT LOCATION
1746 011452      MOV   $BFR+D$LST,TEMP3    ; COPY LINK SERVICE THRESHOLD
1747 011460      CLR   TEMP4              ; INIT LOCATION
1748 011464      MOV   $BFR+D$IPL,TEMP4    ; COPY INPUT PACKET LIMITER
1749 011472      ERRPT$ NMG13A,MSGOUT      ; DISPLAY MORE EXECUTOR INFO
1750
1751          ;
1752          ; DISPLAY ROUTING TYPE
1753          ;
1754 011502      410$: ERRPT$ NMSG8,MSGOUT   ; DISPLAY ROUTING TYPE
1755          ;
1756          ; DISPLAY RECEIVE/TRANSMIT PASSWORDS
1757          ;

```

```

2316 .SBTTL DMXACD - Display X25-ACCESS destinations ;[TD001]
2317 + ;[TD001]
2318 *** DMXACD - Display X25-ACCESS destinations ;[TD001]
2319 ;[TD001]
2320 INPUTS ;[TD001]
2321 RQB - Request Block ;[TD001]
2322 ;[TD001]
2323 OUTPUTS ;[TD001]
2324 Information displayed on user's terminal ;[TD001]
2325 ;[TD001]
2326 DMXACD: ;[TD001]
2327 MOV $PSIPT,R0 ; GET ADDRESS OF PSI HOME BLOCK
2328 ADD #H$RDTE,R0 ; POINT TO RDT LISTHEAD
2329 GETRV R0,#2 ; READ IN LISTHEAD
2330 10$: CALL NXTRDT ; GET NEXT REMOTE DESTINATION BLOCK
2331 BCS 50$ ; BR IF END OF LIST
2332 BIS #F.FND,FLAGS ; Indicate destination found ;[TD001]
2333
2334 MOV #R$NAM+$BFR,R0 ; POINT TO RDT NAME
2335 MOV #TEMP1,R1 ; POINT TO TEMPORARY BUFFER
2336 MOV #6,R2 ; NUMBER OF BYTES IN RDT NAME
2337 20$: MOVB (R0)+,(R1)+ ; STORE RDT NAME
2338 SOB R2,20$ ; ...
2339 30$: CMQB #SPACE,-(R1) ; BACKUP OVER SPACES
2340 BEQ 30$ ; ...
2341 CLRB 1(R1) ; CREATE ASCIZ STRING
2342 ERRT$ ACMS1,MSGOUT ; DISPLAY RDT NAME
2343
2344 MOVB $BFR+$RDAL,R2 ; GET NUMBER OF DIGITS IN DTE ADDRESS
2345 MOV #$BFR+$RDTE,R1 ; POINT TO DTE ADDRESS
2346 MOV #TEMP1,R3 ; POINT TO TEMPORARY BUFFER
2347 CALL BCDASC ; CONVERT DTE ADDRESS TO ASCIZ
2348 TST R$QWN+$BFR ; IS SCOPE GLOBAL?
2349 BEQ 40$ ; BR IF YES
2350 SWTCHI #TEMP2 ; USE TEMP2 AS TEMP INPUT BUFFER
2351 MOV #TEMP2,R1 ; GET ADDRESS OF TEMP INPUT BUFFER
2352 GETRV R$QWN+$BFR,#U.LEN ; READ IN UCB
2353 MOVB U.UNIT(R1),TEMP4 ; SAVE UNIT NUMBER
2354 GETRV U.DCB(R1),#D.LEN ; READ IN DCB
2355 MOV D.NAM(R1),TEMP3 ; SAVE DEVICE NAME
2356 CLRB 105067 162777 ; CREATE ASCIZ STRING
2357 SWTCHI ; $BFR IS INPUT BUFFER
2358 ERRT$ ACMS2A,MSGOUT ; DISPLAY SCOPE AND DTE ADDRESS
2359 BR 10$ ; GET NEXT RDT
2360
2361 40$: ERRT$ ACMS2B,MSGOUT ; DISPLAY SCOPE AND DTE ADDRESS
2362 BR 10$ ; GET NEXT RDT
2363
2364 50$: BITB #F.FND,FLAGS ; WAS AN RDT FOUND?
2365 BNE 70$ ; BR IF YES - DONE
2366 BIT #OP$DST,$OPTON ; SHOW SPECIFIC RDT?
2367 BNE 60$ ; BR IF YES
2368 ERRT$ NOINFO,MSGOUT ; ELSE DISPLAY NO INFORMATION MESSAGE
2369 BR 70$ ; AND EXIT
2370 60$: ERRT$ ACMS3,MSGOUT ; RDT NOT IN SYSTEM
2371 70$: RETURN
2372 ;

```

```
VDIS - VNP SHOW COMMANDS
DMX5S - SHOW MODULE X25-SERVER
```


3403
3404
3405
3406
3407
3408
3409
3410
3411
3412
3413
3414
3415
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
3430

025120 000261
025122 005303
025124 002406
025126 012500
025130 116001 000002
025132 066701 000000G
025140 011101
025142

```

.SBTTL  NXTLIN - GET NEXT LINE
;+
; NXTLIN - GET NEXT LINE
; INPUTS:
;   R3 = NUMBER OF SLT'S REMAINING TO SCAN
;   R5 = ADDRESS OF SLT OF LINE
; OUTPUTS:
;   R0 = SLT ADDRESS
;   R1 = ADDRESS OF DDM PDV
;   R5 = ADDRESS OF SLT OF NEXT LINE IN SYSTEM
; ALL OTHER REGISTERS ARE PRESERVED
;-
NXTLIN: SEC          ; ASSUME ERROR
          DEC         ; DECREMENT COUNT OF NUMBER OF SLT'S
          BLT         ; BR IF NO MORE
          MOV         ; GET SLT ADDRESS
          MOV         ; GET DDM PDV INDEX
          ADD         ; POINT INTO PDV INDEX TABLE
          MOV         ; GET ADDRESS OF DDM PDV
          (R1),R1
10$:     RETURN

```

```

3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949 030042
3950 030056 005704
3951 030060 001012
3952 030062
3953 030072 016600 000012
3954 030076 016601 000010
3955 030102 016602 000006
3956
3957
3958
3959 030106
3960 030112
3961 030122 016600 000012
3962 030126 016601 000010
3963 030132 016602 000006
3964
3965
3966
3967 030136 116003 000015
3968 030142 032760 000400 000000
3969 030150 001006
3970 030152 105760 000014
3971 030156 001403
3972 030160 011604
3973 030162 116403 000001
3974 030166 110367 150642
3975
3976
3977
3978 030172 012705 001062
3979 030176 012704 000641
3980 030202 026127 000000G 015355
  
```

.SBTTL DISCIR - DISPLAY CIRCUIT INFORMATION

```

;+
DISCIR - DISPLAY CIRCUIT INFORMATION
INPUTS:
R0 = CIRCUIT'S SLT ADDRESS
R1 = CIRCUIT'S DDM PDV ADDRESS
R2 = IF MULTIPOINT CIRCUIT -
      TRIB NUMBFR
      IF NOT MULTIPOINT CIRCUIT -
      NO MEANING
R3 = NUMBER OF SLT'S REMAING TO SCAN
R4 = 0 PRINT HEADER INFORMATION ALONG WITH CIRCUIT INFORMATION
      ELSE, PRINT ONLY CIRCUIT INFORMATION
R5 = ADDRESS OF MULTIPOINT FLAG TABLE IF R2 IS POSITIVE

OUTPUTS:
CIRCUIT INFORMATION IS DISPLAYED ON USER'S TERMINAL

ALL REGISTERS ARE PRESERVED
-

DISCIR: SAVRG <R0,R1,R2,R3,R4,R5> ; SAVE R0-R5
      TST R4 ; DISPLAY HEADER?
      BNE 10$ ; BR IF NO
      ERRPT$ MSG1,MSGOUT ; DISPLAY HEADER MESSAGE
      MOV 12(SP),R0 ; RESTORE R0
      MOV 10(SP),R1 ; RESTORE R1
      MOV 6(SP),R2 ; RESTORE R2

; DISPLAY CIRCUIT-ID
10$: CALL FMTCLR ; CREATE THE CIRCUIT-ID
      ERRPT$ MSG2,MSGOUT ; DISPLAY THE CIRCUIT-ID
      MOV 12(SP),R0 ; RESTORE R0
      MOV 10(SP),R1 ; RESTORE R1
      MOV 6(SP),R2 ; RESTORE R2

; GET THE COST
;
      MOVBL L.COST(R0),R3 ; ASSUME COST = CIRCUIT COST
      BIT #LF.BRO,L.FLG(R0) ; ETHERNET CIRCUIT ?
      BNE 20$ ; IF YES - BRANCH
      TSTB L.NSTA(R0) ; MULTIPOINT CIRCUIT?
      BEQ 20$ ; BR IF NO
      MOV (SP),R4 ; GET MULTIPOINT FLAG TABLE
      MOVBL S.COST(R4),R3 ; GET TRIBUTARY COST
20$: MOVBL R3,TEMP1 ; GET COST FOR DISPLAY MESSAGE

; GET CIRCUIT TYPE
30$: MOV #TEMP2,R5 ; STORAGE FOR CIRCUIT TYPE
      MOV #LX25,R4 ; ASSUME X25 CIRCUIT
      CMP Z.NAM(R1),#*RDLM ; IS THIS DLM?
  
```

```

4552          .SBTTL FMTLIN - FORMAT LINE-ID
4553
4554          ;+
4555          FMTLIN - FORMAT A LINE-ID
4556
4557          INPUTS:
4558              R0 = SLT ADDRESS
4559              R1 = ADDRESS OF PDV
4560
4561          OUTPUTS:
4562              TEMP3 = CONTAINS ASCII LINE-ID
4563
4564          ALL REGISTERS PRESERVED
4565
4566          ;
4567          ;
4568
4569          FMTLIN: SAVRG <R0,R1,R2,R3>          ; SAVE REGISTERS
4570                  CLR R3                      ; ASSUME NOT A MUX DEVICE
4571                  BIT #ZF,MUX,Z.FLG(R1)        ; IS IT A MUX DEVICE?
4572                  BEQ 10$                      ; BR IF NO
4573                  INC R3                      ; REMEMBER IT
4574                  MOV Z.NAM(R1),R1             ; GET DDM PROCESS NAME
4575                  MOV #TEMP3,R0               ; STORAGE FOR LINE-ID
4576                  CALL $CSTA                  ; CONVERT NAME TO ASCII
4577                  CMPB ~(R0),#SPACE           ; DELETE TRAILING SPACES
4578                  BEQ 20$
4579                  INC R0                      ; POINT PAST LAST GOOD CHARACTER
4580
4581          ;
4582          ;
4583          ;
4584          ;
4585          ;
4586          ;
4587          ;
4588          ;
4589          ;
4590          ;
4591          ;
4592          ;
4593          ;
4594          ;
4595          ;
4596          ;
4597          ;
4598          ;
4599          ;
4600          ;
4601          ;

```

AALL	000135R	CMSG9	032546R	DSMS1	023340R	D\$JPL	000051	E\$NBS	000020
ACMS1	016320R	COMMA =	000054	DSMS10	024050R	D\$LID	000020	E\$NCR	000034
ACMS2A	016374R	C\$PEC	00C557R	DSMS11	024076R	D\$LNAM	000006	E\$NCS	000036
ACMS2B	016444R	CUNDEC	035300R	DSMS12	024122R	D\$LNK	000000	E\$NIC	000044
ACMS3	016520R	C\$TDEV	005214R	DSMS2A	023406R	D\$LNJM	000014	E\$NLEN	000050
ACNS1	017160R	C\$THX	001575R	DSMS2B	023436R	D\$LST	000047	E\$NLLA	000012
ACNS2	017202R	C\$DTE	000012	DSMS2C	023524R	D\$MAXC	000064	E\$NLNK	000000
ACNS3	017224R	C\$FLG	000015	DSMS3	023576R	D\$MAXH	000066	E\$NML	000040
ACNS4	017252R	C\$GNAM	000016	DSMS4	023622R	D\$MAXV	000070	E\$NMR	000024
ACNS5	017276R	C\$LCN	000010	DSMS5	023652R	D\$MLL	000040	E\$NMS	000030
ACNS6	017346R	C\$LEN	000016	DSMS6	023704R	D\$MNOD	000041	E\$NNDD	000002
ACNS7	017352R	C\$LNK	000000	DSMS7	023742R	D\$NA	000062	E\$NRT	000042
AERR1	003126R	C\$NAM	000002	DSMS8	024006R	D\$NAM	000004	E\$NRTF	000005
AERR2	003162R	C\$NML	000002	DSMS9	024026R	D\$NBEA	000056	E\$NSFG	000010
AKNOWN	000151R	C\$PORT	000014	DTMS1	021000R	D\$NBRA	000054	E\$NTIM	000046
AMSG1	026322R	C\$SCKP=	000000	DTMS2	021070R	D\$NEND=	000054	E\$NUSE	000004
AMSG2	026352R	C\$SORE=	000400	DTMS3	021124R	D\$NLN	000030	E\$SRT	000006
AMSG3	026424R	C\$SRSH=	177564	DV.CCL=	000002	D\$NN	000060	E\$XPR=	000000
AMSG4	026506R	DASH =	000055	DV.COM=	020000	D\$NDD	000010	FF.ADD=	000020
ARAMSK=	***** GX	DEFAULT	000110R	DV.DIR=	000010	D\$OBJ	000003	FF.CIR=	000040
ASPEC	000167R	DF.CRC=	000010	DV.EXT=	000400	D\$OUTJ	000043	FF.CON=	000001
AS\$CHK=	000000	DF.FOC=	000020	DV.F11=	040000	D\$PAL	000026	FF.FIL=	000002
AS\$CPS=	000000	DF.RAN=	000040	DV.LSP=	002000	D\$PRI	000302	FF.HST=	040000
AS\$PRI=	000000	DF.UNK=	000100	DV.MBC=	000400	D\$PRL	000024	FF.LIN=	000010
AS\$TRP=	000000	DISABL	000507R	DV.MNT=	1C0000	D\$RETF	000050	FF.MOD=	000100
A.ACC	000020	DISALT	025666R	DV.MSD=	000100	D\$RNN	000002	FF.MON=	000004
A.LNK	000000	DISCTR	030042R	DV.OSP=	004000	D\$RTMR	000076	FF.MSK=	000077
A.NAM	000002	DISLIN	033074R	DV.PSE=	010000	D\$SEG	000036	FF.PRT=	000200
A.PUD	000010	DISLOO	027630R	DV.REC=	000001	D\$SER	000032	FF.QL=	000370
A.REM	000012	DISNOD	026726R	DV.SDI=	000020	D\$SHI	000020	FF.REM=	100000
A.UCB	000010	DISOBJ	025144R	DV.SQD=	000040	D\$SLD	000016	FLAGS	001574R
BCDASC	035464R	D\$SPRO	026516R	DV.SWL=	001000	D\$SORL	000052	FMTCLR	034334R
BF.TRC=	000004	DISPVC	032602R	DV.TTY=	000004	D\$USL	000025	FMTLIN	034166R
BITNUM=	001204R	DLOEVT	005700R	DV.UMD=	000200	D\$VBL	000032	FNDADD	007276R
BLDLIN	006500R	DLONAM	004756R	D\$ACL	000027	D\$VBL	000032	FNDDBL	024620R
BLDMSK	006366R	DLOSNK	005360R	D\$AMXC	000072	D\$BUG=	177514	FNDPDB	035140R
BLDNOD	006736R	DLOSTA	004716R	D\$AMXH	000074	D\$SL1=	000001	FNDUCB	035212P
BROAD	000652R	DLOTYP	004600R	D\$ANN	000000	D\$SYNC=	000000	F\$SLVL=	000001
CACTIV	000520R	DMDST	022342R	D\$BRPR	000102	D\$SYNM=	000000	F.ADD	000016
CF\$BLK=	000102	DMDTE	020520R	D\$BRTM	000100	D.DSP	000012	F.CEV	000030
CHARAC	000017R	DMGRP	021226R	D\$CUGL	000023	D.LEN =	000036	F.CIR	000024
CHKBIT	007230R	DMXAC	015724R	D\$CUGN	000022	D.LNK	000000	F.CLS	000002
CHKSNK	005236R	DMXACD	015760R	D\$DATL	000030	D.MSK	000014	F.EVT	000004
CIERR1	015156R	DMXACN	016562R	D\$DELF	000045	D.NAM	000004	F.FLG	000014
CKNOWN	000540R	DMXPR	017354R	D\$DELEW	000046	D.PCB	000034	F.FND =	000004
CKPREV	007174R	DMXSS	021762R	D\$DSL	000022	D.UCB	000002	F.LEN	000016
CKTYPE	024544R	DMXSS	024226R	D\$DST	000010	D.UCBL	000010	F.LIN	000020
CKTYP1	024530R	DNMSG1	027352R	D\$DTFL	000031	D.UNIT	000006	F.LNK	000000
CMSG1	032170R	DNMSG2	027366R	D\$END =	000104	D.VCAN=	000002	F.MOD	000022
CMSG11	032572R	DNMSG3	027406R	D\$FLEN	000032	D.VDEB=	177776	F.PREV=	000002
CMSG12	032600R	DNMSG4	027432R	D\$FNB	000034	D.VINI=	000000	F.REM	000026
CMSG2	032220R	DNMSG5	027456R	D\$GLEN	000032	D.VOUT=	000004	F.SIZ=	***** GX
CMSG4	032274R	DNMSG6	027502R	D\$GVBL	000032	D.VPWF=	000006	F.SEV	000004
CMSG5	032344R	DNMSG7	027524R	D\$HIDR	000024	ENABLE	000477R	F.STRT=	000001
CMSG6	032406R	DNMSG8	027552R	D\$HOST	000022	EVENT	000000R	F.URBD=	***** GX
CMSG7	032432R	DNMSG9	027576R	D\$INAC	000044	EVTMSK=	001034R	GCLASS=	001044R
CMSG8	032506R	DNMSG10	027620R	D\$INCT	000042	E\$NBR	000014	GEVMSK=	001746R

SYMBOL	VALUE	REFERENCES
LN.LLO	= 000000	#5-96
LN.DJM	= 000005	#5-96
LN.LQA	= 000004	#5-96
LN.LOO	= 000003	#5-96
LN.OAU	= 000003	#5-96
LN.OFF	= 000001	#5-96
LN.ON	= 000000	#5-96
LN.OOP	= 000004	#5-96
LN.OPE	= 70 001	#5-96
LN.REF	= 000002	#5-96
LN.SER	= 000002	#5-96
LN.STA	= 000017	#5-96
LN.SUB	= 000360	#5-96
LN.TRI	= 000006	#5-96
LOACTV	000256 R	#6-167 12-693
LOCON	000340 R R	#6-176 13-824
LOERR1	004552 R R	12-666 #12-792
LOERR2	004576 R R	12-680 #12-795
LOFIL	000333 R R	#6-175 13-828
LOGIN	000120 R R	#6-138 48-3493
LOGTYP	001676 R R	#6-273 12-690
LOKNWN	000275 R R	#6-168 12-695
LOMON	000323 R R	#6-174 13-820
LOMSG1	004102 R R	12-698 #12-761
LOMSG2	004126 R R	#12-764 14-864
LOMSG3	004174 R R	#12-767
LOMSG4	004432 R R	#12-780 19-1170
LOMSG5	004440 R R	12-750 #12-783
LOMSG6	004464 R R	#12-786 13-839
LOMSG7	004512 R R	#12-789 15-930
LOMS3A	004412 R R	#12-777
LOMS3X	004256 R R	#12-770 18-1061
LOMS3Y	004332 R R	#12-773 18-107
LONOEV	000350 R R	#6-178 19-1121
LOSPEC	000313 R R	#6-169 12-688
LO.ADD	000022	#5-99
LO.ADR	000052	#5-99
LO.CLS	000002	#5-99
LO.CON	000006	#5-99
LO.COP	000010	#5-99
LO.COS	000020	#5-99
LO.CSR	000010	#5-99 5-99
LO.CTL	000032	#5-99
LO.DES	000002	#5-99 5-99 32-2406 41-3137
LO.DTE	000047	#5-99 36-2713
LO.EVT	000004	#5-99
LO.GRO	000024	#5-99 38-2820
LO.HQS	000040	#5-99
LO.HTM	000032	#5-99
LO.INA	000062	#5-99
LO.INC	000060	#5-99
LO.LDT	000046	#5-99 36-2711 36-2715

SYMBOL	VALUE	REFERENCES									
\$PDVNM	= ***** GX	40-3061 41-3135 42-3173									
\$PDVTA	= ***** GX	11-575 61-4815									
\$PFLAG	= ***** GX	11-574 21-1231 35-2661 46-3395 47-3427 52-3905 53-4116 53-4246 55-4428									
\$PSIPT	= ***** GX	55-4444 61-4814 61-4827 33-2426 34-2546 38-2821 40-2938 40-3028 41-3138									
\$QUAL	= ***** GX	27-1952 27-2005 27-2094 31-2327 33-2428 34-2545 35-2633 37-2739 40-2925									
\$RADDR	= ***** GX	54-4341 27-1946 27-2000 27-2088 31-2327 33-2428 34-2545 35-2633 37-2739 40-2925									
\$RBLCK	= ***** GX	42-3166 9-389 10-466 10-470 11-577 11-579 12-686 12-691 12-702 12-748									
\$RDBNM	= ***** GX	13-830 26-1442 26-1447 27-1929 27-1931 27-2026 27-2033 27-2042 28-2161									
\$RDBSZ	= ***** GX	28-2163 28-2199 28-2236 22-1298 22-1308 25-1397 25-1408 26-1439 26-1484 26-1524									
\$RDBTH	= ***** GX	*10-461 *18-1023 *26-1630 *26-1684 *26-1706 *26-1717 *26-1762 *26-1763 *27-1949 *27-2003									
\$RSIZE	= ***** GX	*27-2091 *31-2329 *31-2352 *31-2354 *33-2430 *34-2553 *34-2570 *35-2635 *37-2741 *37-2741									
\$SAVAL	= ***** GX	*37-2767 *39-2880 *39-2894 *40-2926 *42-3168 *42-3180 *44-3280 *45-3364 *49-3577 *49-3577									
\$SAVRG	= ***** GX	*49-3578 *53-4021 *53-4025 *53-4049 *53-4053 *53-4080 *53-4084 *53-4208 *53-4258 *53-4258									
\$SDBNM	= ***** GX	*53-4262 *55-4473 *58-4682 *59-4733 *59-4743 *59-4749 *60-4778 *60-4782 *62-4848 *62-4848									
\$SLTMA	= ***** GX	*34-2566 *39-2890 *39-2893 *53-4015 *53-4020 *53-4043 *53-4048 *53-4074 *53-4079 *53-4259									
\$SLTNM	= ***** GX	8-357 8-357 26-1716 8-357 8-357 12-727 12-728 12-729 54-4333 66-5000									
\$TIOUT	= ***** GX	18-1015 44-3276 8-353 8-353 21-1227 27-1926 28-2158 35-2658 52-3901 53-4251									
\$TYPE	= ***** GX	27-1927 28-2159 53-4194 27-1927 28-2159 53-4194 27-1927 28-2159 53-4194 27-1927 28-2159 53-4194									
\$VFLAG	= ***** GX	43-3217 43-3221 43-3224 39-2866 59-4718 8-351 8-352 8-352 8-352									
\$	= 177777 GX	12-673 26-1527 26-1543 39-2866 59-4718 8-351 8-352 8-352 8-352									
		6-236 6-236 #6-236 #6-237 8-351 8-356 #8-351 8-352 8-352									
		#8-352 #8-353 8-355 8-355 #8-355 8-356 #8-356 8-356 #8-356									
		8-362 8-362 #8-362 #8-363 9-434 9-434 #9-434 9-435 #9-435									
		10-547 #10-547 #10-548 10-550 10-550 #10-550 10-551 11-640 11-640									
		#11-640 #11-641 12-760 12-760 #12-760 12-761 12-763 12-763 #12-763									
		#12-764 12-766 12-766 #12-766 12-767 12-769 12-769 #12-769									
		12-772 12-772 #12-772 12-773 12-776 12-776 #12-776 12-777 12-777									
		12-779 #12-779 #12-780 12-782 12-782 #12-782 12-783 12-785 12-785									
		#12-785 #12-786 12-788 12-788 #12-788 12-789 12-791 12-791 #12-791									
		#12-792 12-794 12-794 #12-794 12-795 26-1830 26-1830 #26-1830									
		26-1833 26-1833 #26-1833 26-1834 26-1836 26-1836 #26-1836									
		26-1839 #26-1839 #26-1840 26-1842 26-1842 #26-1842 26-1843 26-1845 26-1845									
		#26-1845 26-1846 26-1848 26-1848 #26-1848 26-1849 26-1851 26-1851									
		26-1852 26-1852 #26-1852 26-1853 26-1855 26-1855 #26-1855									
		#26-1856 26-1857 26-1859 26-1859 #26-1859 26-1860 26-1862 26-1862									
		#26-1863 26-1865 26-1865 #26-1865 26-1866 26-1868 26-1868 #26-1868									
		26-1871 26-1871 #26-1871 26-1872 26-1874 26-1874 #26-1874									
		26-1877 #26-1877 #26-1878 26-1880 26-1880 #26-1880 26-1881 26-1884 26-1884									
		#26-1884 26-1885 26-1885 #26-1885 26-1886 26-1886 #26-1886 26-1887 26-1887									
		26-1889 #26-1889 26-1890 26-1890 #26-1890 26-1891 26-1893 26-1893 #26-1893									

```

LOCAL DATA
113          .SBTTL  LOCAL DATA
114
115 000000          .PSECT  DATA,D
116
117
118 000000 054134 003310  ACPNAM: .RAD50 /NETACP/
119 000004 115443 003310  X2SNAM: .RAD50 /X2SACP/
120 000010 055251 054374  NTINAM: .RAD50 /NTINIT/
121 000014 055251 054374  NTI11S: .RAD50 /NTINIT/ ; IF 11S NAME IS DIFFERENT, USE IT HERE
122 000020 021265 055254 057330  DEFFIL: .RAD50 /EVENTLOG/ ; DEFAULT FILE NAME
123 000026 075273  DEFTYP: .RAD50 /SYS/ ; DEFAULT FILE TYPE
124 000030 000000  DEFEVER: .WORD 0 ; DEFAULT VERSION NUMBER
125 000032 114 102  DEFDEV: .ASCII /LB/ ; DEFAULT DEVICE NAME
126 000034 000000  DEFUNT: .WORD 0 ; DEFAULT UNIT NUMBER
127 000036 006 001  DEFUIC: .BYTE 6,1 ; DEFAULT UIC
128 000020 051646 131574  DEFMON: .RAD50 /MON.../ ; LENGTH OF DEFAULT FILE NAME
129 000040 103 117  DEFCON: .ASCII /CO/ ; DEFAULT MONITOR NAME
130 000044 000000  DEFLEN: .WORD 0 ; DEFAULT CONSOLE NAME
131 000046 000030  DFLEN = .-DEFFIL ; LENGTH OF ALL OF DEFAULT NAMES FOR EVENT LOGGING
132 000050 000001  CLKQB: .BLKW C.LGTH ; CLOCK QUEUE ENTRY BUFFER
133 000050 000001  TICKS = 1 ; NUMBER OF TICKS TO WAIT FOR NTINIT TO RUN
134 000001  SER = 1 ; SERVICE FLAG
135 000001  CHAN: .BLKW 1 ; CHANNEL NUMBER
136 000070  PDVX: .BLKW 1 ; PDV INDEX
137 000072  CLASS: .BLKW 1 ; EVENT CLASS
138 000074  TMPMSK: .BLKW 4 ; TEMPORARY STORAGE FOR EVENT MASKS
139 000076  GMASK: .BLKW 4 ; GLOBAL EVENT MASK
140 000106  GCLASS: .BLKW 1 ; GLOBAL EVENT CLASS
141 000116  EVQUAL: .BLKW 1 ; EVENT FILTER BLOCK QUALIFIER FLAGS
142 000120  PLBFLG: .BLKB 1 ; PLB FLAG, =1 IF PLB DATA BASE IS PRESENT
143 000122  LCLFLG: .BLKB 1 ; Local state flag, used in ALPLB
144 000123  SNAPX: .BLKB 1 ; SNA PDV index, used in $VENA
145 000124  DLXPX: .BLKB 1 ; DLX PDV index, used in $VENA
146 000125  .EVEN
147
148 000126 000000  SERFLG: .WORD 0 ; CIRCUIT SERVICE FLAG WORD
149 000130  SINKND: .BLKW 1 ; SINK NODE ADDRESS FOR EVENT LOGGING
150 000132  REMOTE: .BLKW 1 ; REMOTE NODE ADDRESS FOR EVENT LOGGING
151 000134  FILNAM: .BLKW 3 ; FILE NAME (RAD50)
152 000142  FILTYP: .BLKW 1 ; FILE TYPE (RAD50)
153 000144  FILVER: .BLKW 1 ; VERSION NUMBER (BINARY)
154 000146  FILDEV: .BLKW 1 ; DEVICE NAME (ASCII)
155 000150  FILUNT: .BLKW 1 ; DEVICE UNIT NUMBER (BINARY)
156 000152  FILUIC: .BLKW 1 ; UIC (BINARY)
157 000154  MONNAM: .BLKW 2 ; MONITOR NAME (RAD50)
158 000160  CONNAM: .BLKW 2 ; CONSOLE NAME (ASCII AND BINARY)
159 000164  CURUNM: .BLKW 1 ; UNMAPPED ADDRESS OF CURRENT BLOCK
160 000166  PRVUNM: .BLKW 1 ; UNMAPPED ADDRESS OF PREVIOUS BLOCK IN LIST
161 000170  FLTBUF: .BLKB F.CEV+10 ; STORAGE FOR EVENT FILTER BLOCK
162  .EVEN
163
164 ; SINK TYPE TABLE (USED FOR 'KNOWN LOGGING')
165
166 000230 000001  SNKTP: .WORD FF.CON ; CONSOLE SINK TYPE
167 000232 000002  .WORD FF.FIL ; FILE SINK TYPE
168 000234 000004  .WORD FF.MON ; MONITOR SINK TYPE
169 000236 000000  .WORD 0 ; END OF TABLE

```

```

649 .SBTTL $VDEA - SET DEAD POLLING RATE
650 ;+
651 ; **-$VDEA-SET DEAD POLLING RATIO FOR A LINE
652 ;
653 ; THIS ROUTINE IS CALLED TO CHANGE THE DEAD POLLING RATE
654 ; FOR A LINE.
655 ;
656 ; INPUTS:
657 ; $SLTA = SLT ADDRESS
658 ;
659 ; OUTPUTS:
660 ; C-BIT = SUCCESS/FAILURE
661 ;
662 ; -
663
664 001430 $VDEA::
665 001430 016701 000000G MOV $SLTA,R1 ; GET SLT ADDRESS
666 001434 032761 000000G 000000G BIT #LF.MTP,L.FLG(R1) ; IS THIS A MULTI-POINT LINE ?
667 001442 001412 BEQ 101$ ; IF EQ, NO - PARAMETER NOT APPLICABLE
668
669 001444 CALL FNDLIN ; FIND THE DCP LINE TABLE
670 001450 103406 BCS 10$ ; IF CS, ERROR DETECTED
671 001452 001412 BEQ 102$ ; IF EQ, NO LINE TABLE
672 001454 116767 000024G 000047G MOVB R0B+L0.MDE,$BFR+L.PLD ; SET THE NEW POLLING RATE
673 001462 PUTAD ; WRITE BACK THE LINE TABLE
674 001466 10$: RETURN ; BACK TO USER STATE
675 ;
676 ; ERROR CONDITIONS
677
678 001470 101$: ERRPT$ ERR31,$EROUT ; LINE IS NOT MULTIPOINT
679 001500 102$: ERRPT$ ERR34,$EROUT ; INVALID PARAMETER, MULTIPOINT DEAD

```


VENA - VNP ENABLE/DISABLE LINES MACRO V05.03b Wednesday 11-Sep-85 12:06 Page 25
 DSBLIN - DISABLE A LINE

```

1156
1157      *** - $SNOD - PROCESS SET NODE REQUESTS
1158      *** - $CNOD - PROCESS CLEAR NODE REQUESTS
1159
1160      INPUTS:
1161      $OPTON = OPTIONS SELECTED
1162
1163      OUTPUTS:
1164      NONE.
1165
1166      REGISTERS:
1167      ALL REGISTERS DESTROYED.
1168
1169      -
1170      .ENABLE LSB
1171 003460 $SNOD::
1172 003460 012705 003660' MOV #SNODTB,R5 ; GET ADDRESS OF SET NODE TABLE
1173 003464 000402 BR 5$ ; AND CONTINUE
1174 003466 $CNOD::
1175 003466 012705 003714' MOV #CNODTB,R5 ; GET ADDRESS OF CLEAR NODE TABLE
1176 003472 5$: CALL CHKEXE ; IS IT EXECUTOR NODE ?
1177 003476 001005 BNE 10$ ; IF NE, NO
1178
1179 ; CHECK FOR INVALID EXECUTOR OPTIONS
1180
1181 003500 032767 000044 000000G BIT #OP$NAM!OP$LNE,$OPTON ; WAS NODE NAME OR LINE ID SPECIFIED ?
1182 003506 001045 BNE 101$ ; IF NE, YES .. INVALID OPTION(S)
1183 003510 000420 BR 20$ ; ELSE, CONTINUE
1184
1185 ; CHECK FOR INVALID NODE OPTIONS
1186
1187 003512 032767 017400 000000G 10$: BIT #OP$INC!OP$OUT!OP$INA!OP$RET!OP$SEG,$OPTON ; ILLEGAL OPTION ?
1188 003520 001040 BNE 101$ ; IF YES - BRANCH
1189 003522 032767 000171 000000G BIT #OP$HOS!OP$RPA!OP$TPA!OP$VER!OP$EST,$OPTON ; ILLEGAL OPTION ?
1190 003530 001034 BNE 101$ ; IF NE, YES .. INVALID OPTION
1191 003532 032767 000040 000000G BIT #OP$LNE,$OPTON ; WAS LINE ID SPECIFIED ?
1192 003540 001404 BEQ 20$ ; IF EQ, NO .. OKAY
1193 003542 032767 000004 000000G BIT #OP$NAM,$OPTON ; ELSE, MAKE SURE NAME NOT PRESENT !
1194 003550 001024 BNE 101$ ; IF NE, NAME PRESENT ALSO .. ERROR
1195
1196 003552 016704 000000G 20$: MOV $OPTON,R4 ; GET OPTIONS WORD
1197 003556 000241 CLC ; MAKE SURE CARRY IS CLEAR
1198 003560 006004 30$: ROR R4 ; GET NEXT OPTION
1199 003562 103006 BCC 40$ ; IF CC, NOT PRESENT
1200 003564 010446 MOV R4,-(SP) ; SAVE R4
1201 003566 010546 MOV R5,-(SP) ; SAVE R5
1202 003570 CALL @R5 ; ELSE, EXECUTE REQUEST
1203 003574 012605 MOV (SP)+,R5 ; RESTORE R5
1204 003576 012604 MOV (SP)+,R4 ; RESTORE R4
1205 003600 005725 40$: TST (R5)+,R4 ; SKIP TO NEXT REQUEST
1206 003602 001406 BEQ 50$ ; IF EQ, END OF TABLE
1207 003604 005715 TST (R5) ; CHECK FOR MINUS
1208 003606 002002 BGE 45$ ; IF NO - BRANCH
1209 003610 006004 ROR R4 ; SKIP ONE BIT
1210 003612 005725 TST (R5)+ ; SHIP TO NEXT REQUEST
1211 003614 005704 45$: TST R4 ; ANY MORE ?
1212 003616 001360 BNE 30$ ; IF NE, YES

```

VENA - VNP ENABLE/DISABLE LINES MACRO V05.03b Wednesday 11-Sep-85 12:06 Page 25-1
 DSBLIN - DISABLE A LINE

VENA - VNP ENABLE/DISABLE LINES MACRO V05.03b Wednesday 11-Sep-85 12:06 Page 33
CLRBIT - CLEAR LOOPBACK BIT

```

1724                                     .SBTTL CLRBIT - CLEAR LOOPBACK BIT
1725                                     ;+
1726                                     ;**~CLRBIT~CLEAR LOOPBACK BIT
1727                                     ;
1728                                     ;INPUTS:
1729                                     ;R2 = CHANNEL NO.
1730                                     ;
1731                                     ;OUTPUTS:
1732                                     ;C-BIT = SUCCESS/FAILURE
1733                                     ;R2,R3 = DESTROYED
1734                                     ;
1735                                     ;-
1736
1737 005724                               CLRBIT:
1738 005724                               CALL    GTSLT                ; GET SLT ADDRESS
1739 005730 103420                       BCS     20$                ; IF CS, NO XPT
1740 005732 105762 000000G               TSTB     L,NSTA(R2)        ; IS THIS LINE MULTIPOINT ?
1741 005733 001004                       BNE     10$                ; IF NE, YES
1742 005740 042762 000000G 000000G     BIC     #L.LPB,L.FLG(R2)    ; ELSE, CLEAR THE LOOP BACK BIT
1743 005746 000411                       BR      20$                ; AND RETURN
1744 005750 006303                       10$:  ASL     R3                ; POINT TO STATION INFO IN SLT
1745 005752 006303                       ASL     R3                ; ...
1746 005754 060302                       ADD     R3,R2                ; ...
1747 005756 062702 000000G             ADD     #L.MPF,R2            ; ...
1748 005762 142762 000000G 000000G     BICB    #SF.LPB,S.FLG(R2)    ; AND CLEAR THE LOOPBACK BIT
1749 005770 000241                       CLC                      ; INDICATE SUCCESS
1750 005772                               20$:  RETURN

```

VENA - VNP ENABLE/DISABLE LINES MACRO V05.03b Wednesday 11-Sep-85 12:06 Page 34
CLRBIT - CLEAR LOOPBACK BIT

```

2287          .SBTTL $VALI - SET/CLEAR ALIAS
2288
2289      *** - $VALI - SET/CLEAR ALIAS
2290
2291      INPUTS:
2292          $OPTON = OPTIONS WORD
2293          REQUEST BLOCK CONTAINS DATA
2294
2295      OUTPUTS:
2296          NONE.
2297
2298      -
2299
2300          .ENABL LSB
2301      $VALI::
2302          MOV    #R0B+LR.ALI,R1      ; GET THE ALIAS NAME STRING
2303          BIT    #CM$SET,$CMAND      ; IS THIS A SET COMMAND ?
2304          BNE    20$                 ; IF NE, YES
2305
2306      ; CLEAR REQUEST
2307          BIT    #QF$ALL!QF$KNO,$QUAL ; IS THIS FOR KNOWN OR ALL NODES ?
2308          BEQ    3$                   ; IF EQ, NO .. CONTINUE
2309          JMP    VALI                 ; ELSE, REMOVE THOSE ALIASES
2310
2311          3$: BIT    #OP$SCO,$OPTON    ; WAS THE SCOPE GIVEN ?
2312          BNE    4$                   ; IF NE, YES .. OKAY
2313          ERRPT$ ERR26,$EROUT        ; ELSE, SCOPE PARAMETER MISSING
2314
2315          4$: CALL   $FNALI            ; TRY TO FIND THE ALIAS
2316          BCC    REMALI              ; IF CC, FOUND .. OKAY
2317          JMP    101$               ; ELSE, ERROR
2318
2319      REMALI: MOV    R1,R3            ; SAVE ADDRESS OF PREVIOUS ENTRY
2320              MOV    $BFR,R4         ; SAVE ADDRESS OF NEXT ENTRY
2321              MOV    #BFR,R1         ; POINT AT START OF BUFFER
2322              ADD    #A,ACC,R1        ; POINT TO ACCOUNTING INFO.
2323              MOV    (R1)+,R2         ; GET LENGTH OF FIELD
2324              ADD    R2,R1            ; POINT TO NEXT FIELD
2325              MOV    (R1)+,R2         ; GET LENGTH OF FIELD
2326              ADD    R2,R1            ; POINT TO NEXT FIELD
2327              MOV    (R1)+,R2         ; GET LENGTH OF LAST FIELD
2328              ADD    R2,R1            ; POINT TO END OF ALIAS BLOCK
2329              SUB    #BFR,R1         ; CALCULATE LENGTH OF ALIAS BLOCK
2330              MOV    R1,-($P)         ; SAVE LENGTH OF BLOCK
2331
2332      ; REMOVE ENTRY FROM LIST
2333
2334          CMP    $DECP,R3            ; IS PREVIOUS ALIAS THE LISTHEAD ?
2335          BNE    10$                 ; D$ANN IF FIRST WORD OF HOME BLOCK
2336          BNE    10$                 ; IF NE, NO
2337
2338          SWTCH  #HMBUF              ; USE TEMPORARY BUFFER
2339          GETRV  $DECP,4,$END        ; READ DECNET HOME BLOCK
2340          SWTCH  $DECP,4,$END        ; RESET TEMPORARY BUFFER
2341
2342          MOV    R4,$HMBUF+D$ANN      ; ELSE, SET NEW NEXT
2343

```

```

2841 013274 020500      CMP      R5,R0      ; END OF STRING?
2842 013276 101530      BLOS     110$      ; BR IF YES
2843 013300 122710 000133  CMPB    #'L',(R0)    ; WAS A UIC SPECIFIED?
2844 013304 001017      BNE      50$      ; BR IF NO
2845 013306 005200      INC       R0      ; POINT PAST LEFT BRACKET
2846 013310      CALL    $C0TB    ; CONVERT USER GROUP CODE
2847 013314 022702 000054  CMP      #'<,>,R2    ; SUCCESSFUL CONVERSION?
2848 013320 001121      BNE      120$    ; BR IF NO
2849 013322 110167 000153'  MOVB     R1,FILUIC+1 ; STORE USER GROUP CODE
2850 013326      CALL    $C0TB    ; CONVERT USER MEMBER CODE
2851 013332 022702 000133'  CMP      #'J',R2    ; SUCCESSFUL CONVERSION?
2852 013336 001112      BNE      120$    ; BR IF NO
2853 013340 110167 000152'  MOVB     R1,FILUIC  ; STORE USER MEMBER CODE
2854
; CHECK FILE NAME
2855
50$: 013344 020500      CMP      R5,R0      ; END OF STRING?
2856 013346 101504      BLOS     110$      ; BR IF YES
2857 013350 122720 000056  CMPB    #'',(R0)+    ; ANY FILE NAME SPECIFIED?
2858 013354 001437      BEQ      80$      ; BR IF NO
2859 013356 005300      DEC       R0      ; POINT TO BEGINNING OF FILE NAME
2860 013360 012704 000134'  MOV      #FILNAM,R4 ; POINT TO FILE NAME SAVE AREA
2861 013364 005014      CLR      (R4)    ; INITIALIZE FILE NAME
2862 013366 005064 000002  CLR      2(R4)      ; ...
2863 013372 005064 000004  CLR      4(R4)      ; ...
2864 013376 012703 000003  MOV      #MXFILN,R3 ; MAXIMUM NUMBER OF WORDS FOR FILE NAME
2865 013402 005001      CLR      R1      ; PERIOD NOT VALID
2866 013404      CALL    $CAT5    ; CONVERT NAME TO RAD50
2867 013410 103011      BCC      70$      ; BR IF SUCCESSFUL CONVERSION
2868 013412 020500      CMP      R5,R0      ; END OF STRING?
2869 013414 101405      BLOS     65$      ; BR IF YES
2870 013416 020227 000056  CMP      R2,#'.'    ; VALID TERMINATOR?
2871 013422 001060      BNE      120$    ; BR IF NO
2872 013424 010124      MOV      R1,(R4)+  ; STORE FILE NAME
2873 013426 000412      BR       80$      ; NOW CHECK FILE TYPE
2874 013430 010124 65$:  MOV      R1,(R4)+  ; STORE FILE NAME
2875 013432 000452      BR       110$    ; AND EXIT
2876 013434 010124 70$:  MOV      R1,(R4)+  ; STORE FILE NAME
2877 013436 005303      DEC       R3      ; MORE TO CONVERT?
2878 013440 003360      BGT      60$      ; BR IF YES
2879 013442 020500      CMP      R5,R0      ; END OF STRING?
2880 013444 101445      BLOS     110$      ; BR IF YES
2881 013446 122027 000056  CMPB    (R0)+,#'.'  ; VALID TERMINATOR?
2882 013452 001044      BNE      120$    ; BR IF NO - TOO MANY CHARACTERS
2883
; CHECK FILE TYPE
2884
80$: 013454 020500      CMP      R5,R0      ; END OF STRING?
2885 013456 101440      BLOS     110$      ; BR IF YES
2886 013460 122720 000073  CMPB    #'',(R0)+    ; ANY FILE TYPE SPECIFIED?
2887 013464 001424      BEQ      100$    ; BR IF NO
2888 013466 005300      DEC       R0      ; POINT TO START OF FILE TYPE STRING
2889 013470 005301      CLR      R1      ; PERIODS NOT VALID
2890 013472      CALL    $CAT5    ; CONVERT TO RAD50
2891 013476 103010      BCC      90$      ; BR IF SUCCESS
2892 013500 020500      CMP      R5,R0      ; END OF STRING?
2893 013502 101406      BLOS     90$      ; BR IF YES

```

```

1173                                     .SBTTL BLDMSK - BUILD EVENT MASK
1174
1175                                     ;+
1176                                     ;
1177                                     ; BLDMSK - BUILD EVENT MASK FOR A FILTER BLOCK WITH A QUALIFIER WHICH
1178                                     ; HAS A GLOBAL FILTER BLOCK ASSOCIATED WITH IT
1179                                     ;
1180                                     ; INPUTS:
1181                                     ; R3 - ADDRESS OF FILTER BLOCK
1182                                     ;
1183                                     ; OUTPUTS:
1184                                     ; EVTMSK - EVENT MASK
1185                                     ; R3 - ADDRESS OF FILTER BLOCK
1186                                     ;
1187                                     ; -
1188
1189 006366 016767 172454 172440 BLDMSK: MOV GEVMSK+0,EVTMSK+0 ; SAVE GLOBAL FILTER MASKS
1190 006374 016767 172450 172434 MOV GEVMSK+2,EVTMSK+2 ; ...
1191 006402 016767 172444 172430 MOV GEVMSK+4,EVTMSK+4 ; ...
1192 006410 016767 172440 172424 MOV GEVMSK+6,EVTMSK+6 ; ...
1193 006416 046367 000030 172410 BIC F.CEV+0(R3),EVTMSK+0 ; CLEAR OUT CLEAR EVENT MASKS
1194 006424 046367 000032 172404 BIC F.CEV+2(R3),EVTMSK+2 ; ...
1195 006432 046367 000034 172400 BIC F.CEV+4(R3),EVTMSK+4 ; ...
1196 006440 046367 000036 172374 BIC F.CEV+6(R3),EVTMSK+6 ; ...
1197 006446 056367 000004 172360 BIS F.SEV+0(R3),EVTMSK+0 ; SET EVENTS FROM SET EVENT MASKS
1198 006454 056367 000006 172354 BIS F.SEV+2(R3),EVTMSK+2 ; ...
1199 006462 056367 000010 172350 BIS F.SEV+4(R3),EVTMSK+4 ; ...
1200 006470 056367 000012 172344 BIS F.SEV+6(R3),EVTMSK+6 ; ...
1201 006476 RETURN

```

```

1758 011512 012700 114224      MOV    #*RXPT,R0      ; GET XPT'S PDV ADDRESS
1759 011516      CALL   FNDPDV      ;
1760 011522 103443      BCS    430$      ; BR IF NONE
1761 011524      SWITCH      ; SET DEFAULT BUFFER ($BFR)
1762 011532      GETRV    Z,DAT(R0),#N$XLEN      ; GET XPT'S DATA BASE
1763 011532      GETRV    N$VER+2+$BFR,#V$LEN      ; GET PASSWORD DESCRIPTOR
1764 011572 032767 100000 000000G  BIT    #VF$RCV,V$FLG+$BFR      ; RECEIVE PASSWORD VALID?
1765 011600 001404      BEQ    420$      ; BR IF NO
1766 011602      ERRPT$   NMSG9,MSGOUT      ; ELSE, DISPLAY RECEIVE PASSWORD
1767 011612 032767 040000 000000G 420$: BIT    #VF$XMT,V$FLG+$BFR      ; TRANSMIT PASSWORD VALID?
1768 011620 001404      BEQ    430$      ; BR IF NO
1769 011622      ERRPT$   NMSG10,MSGOUT      ; ELSE, DISPLAY TRANSMIT PASSWORD
1770
1771      ; DISPLAY VERIFICATION STATE
1772
1773 011632 012701 001062' 430$: MOV    #TEMP2,R1      ; STORAGE FOR VERIFICATION STATE STRING
1774 011636 012700 000064'      MOV    #DN,R0      ; ASSUME IT IS ON
1775 011642      CALL   FNDUCB      ; GET UCB FOR NS: DEVICE
1776 011646 032767 100000 000014G  BIT    #NF$ACC,$BFR+U.CW3      ; IS IT ON?
1777 011654 001002      BNE    440$      ; BR IF YES
1778 011656 012700 000077'      MOV    #OFF,R0      ; ELSE, GET OFF STRING
1779 011662 112021 440$: MOV    (R0)+,(R1)+      ; STORE VERIFICATION STATE STRING
1780 011664 001376      BNE    440$      ;
1781 011666      ERRPT$   NMSG11,MSGOUT      ; DISPLAY VERIFICATION STATE
1782 011676 000452      BR     520$      ; FINISHED
1783
1784      ; SHOW LOOP NODES
1785
1786 011700 012700 000376' 450$: MOV    #NLOOP,R0      ; LOOP TYPE COMMAND
1787 011704 112021 460$: MOV    (R0)+,(R1)+      ; STORE TYPE STRING
1788 011706 001376      BNE    460$      ;
1789 011710 005004      CLR     R4      ; INDICATE HEADER TO BE PRINTED
1790
1791 011712      470$: CALL   NXTBLK      ; READ IN NEXT REMOTE NAME BLOCK
1792 011716 103415      BCS    490$      ; BR IF END OF LIST
1793 011720 005763 000012      TST    R.FLAG(R3)      ; IS THIS A LOOPBACK NODE?
1794 011724 100372      BPL    470$      ; BR IF NO
1795 011726 005704      TST    R4      ; DISPLAY HEADER?
1796 011730 001004      BNE    480$      ; BR IF NO
1797 011732      ERRPT$   NMSG1,MSGOUT      ; DISPLAY HEADER
1798 011742      480$: CALL   DISLOO      ; DISPLAY LOOPBACK NODE INFORMATION
1799 011746 005204      INC     R4      ; INDICATE HEADER ALREADY PRINTED
1800 011750 000760      BR     470$      ; GET NEXT NAME BLOCK
1801 011752 005704      490$: TST    R4      ; ANY INFO DISPLAYED?
1802 011754 001423      BEQ    520$      ; BR IF NO
1803 011756 000416      BR     510$      ;
1804
1805      ; SHOW DLX EXECUTOR INFORMATION
1806
1807      ; FIRST - "EXECUTOR NODE"
1808
1809 011760 500$: ERRPT$   NMSG14,MSGOUT      ; DISPLAY "EXECUTOR NODE"
1810
1811      ; FIND STATE OF EXECUTOR AND DISPLAY
1812
1813      ;
1814 011770      CALL   GTSTJD      ; GET STATE

```

```
2373 ; MESSAGES
2374 ;
2375 016304 ERMSG$ < Name = %A>
2376 016316 ERBLK$ ACMS1,<TEMP1>
2377
2378 016324 ERMSG$ < DTE = %A, Scope = Terminal %A%B:%N>
2379 016372 ERBLK$ ACMS2A,<TEMP1,TEMP3,TEMP4>
2380
2381 016404 ERMSG$ < DTI = %A, Scope = Global%N>
2382 016442 ERBLK$ ACMS2B,<TEMP1>
2383
2384 016450 ERMSG$ < Remote destination not in system%N>
2385 016516 ERBLK$ ACMS3
```

```

2893 022160 016067 000000G 000000G      MOV      Z,DSP(R0),%RBLCK      ; GET PROCESS MAPPING BIAS
2894 022166      GETAD      R1,%Z$LEN      ; READ IN LINE TABLE
2895 022204 016767 000014G 156622      MOV      Z,%CTIM+%BFR,TEMP1      ; GET COUNTER TIMER
2896 022212      ERRT%      X5MS1,MSGOUT      ; DISPLAY INFO TO USER'S TERMINAL
2897 022222      60%:      RETURN
2898      ;
2899      ; ERRORS
2900      ;
2901 022224      ERMSG$ <      NW process not in system%N>
2902 022262      ERBLK$ X5ER1
2903      ;
2904      ; MESSAGES
2905      ;
2906 022266      ERMSG$ <      Stat% = %A, Counter timer = %F%N>
2907 022332      ERBLK$ X5MS1,<TEMP2,TEMP1>
  
```


.SBTTL DISOBJ - DISPLAY OBJECT INFO

DISOBJ - DISPLAY OBJECT INFORMATION

INPUTS:

R3 = ADDRESS OF OBJECT TABLE
R4 = 0 IF HEADER IS TO BE PRINTED ALONG WITH OBJECT INFORMATION
ELSE, ONLY THE OBJECT INFORMATION IS PRINTED
IF HEADER IS TO BE PRINTED:
TEMP1 = SECOND ASCIZ STRING OF HEADER MESSAGE
TEMP2 = FIRST ASCIZ STRING OF HEADER MESSAGE

OUTPUTS:

THE OBJECT INFORMATION IS DISPLAYED ON THE USER'S TERMINAL

R3,R4,R5 ARE PRESERVED

DISOBJ: TST R4 ; PRINT HEADER?
BNE 10\$; BR IF NO
ERRPT\$ MSG1,MSGOUT ; PRINT HEADER

DISPLAY OBJECT TYPE

10\$: MOVB 0.TYP(R3),TEMP1 ; SAVE OBJECT TYPE
ERRPT\$ MSG2,MSGOUT

SAVE NAME TO DISPLAY

MOV 0.NAM(R3),TEMP2 ; SAVE RAD50 OBJECT NAME
MOV 0.NAM+2(R3),TEMP2+2 ; ...

DISPLAY ACCESS LEVEL INFORMATION

MOV #TEMP1,R1 ; GET AREA TO STORE VERIFICATION STRING
MOV #ON,R0 ; ASSUME ACCESS LEVEL IS ON
MOVB 0.FLG(R3),R2 ; GET FLAGS BYTE
BIC #^C<7>,R2 ; CLEAN OUT FLAG BITS
CMPB #OF\$ON,R2 ; IS IT ON?
BEQ 20\$; BR IF YES
MOV #INSPT,R0 ; ASSUME INSPECT
CMPB #OF\$INS,R2 ; IS IT INSPECT?
BEQ 20\$; BR IF YES
MOV #OFF,R0 ; MUST BE OFF
MOVB (R0)+(R1)+,R2 ; STORE ACCESS LEVEL
BNE 20\$; ...
TSTB 0.TYP(R3) ; OBJECT TYPE 0?
BNE 30\$; BR IF NO
ERRPT\$ MSG3,MSGOUT ; DISPLAY VERIFICATION
SR 40\$;
30\$: ERRPT\$ MSG4,MSGOUT ; DISPLAY NAME AND VERIFICATION

SAVE NUMBER OF COPIES

40\$: MOVB 0.MXC(R3),TEMP2 ; SAVE NUMBER OF COPIES

```

3981 030210 001424 BEQ 50$ ; BR IF YES
3982 030212 105760 000014 TSTB L,NSTA(R0) ; MULTIPOINT CIRCUIT?
3983 030216 001411 BEQ 40$ ; BR IF NO - BRANCH
3984 030220 012704 000652' MOV #BROAD,R4 ; ASSUME ETHERNET DEVICE ?
3985 030224 032760 000400 000000 BIT #LF,BRO,L.FLG(R0) ; BROADCAST DEVICE ?
3986 030232 001013 BNE 50$ ; IF YES - BRANCH
3987 030234 012704 000567' MOV #LCNTRL,R4 ; IF NO - DDCMP MASTER
3988 030240 000410 BR ; CONTINUE
3989 030242 012704 000621' 40$: MOV #LTRIB,R4 ; ASSUME TRIBUTARY
3990 030246 032760 000020 000000 BIT #LF,MTP,L.FLG(R0) ; MULTIPOINT CIRCUIT?
3991 030254 001002 BNE 50$ ; BR IF YES
3992 030256 012704 000605' MOV #LPOINT,R4 ; ELSE IT MUST BE POINT-TO-POINT
3993 030262 112425 50$: MOVB (R4)+(R5)+ ; STORE CIRCUIT TYPE STRING
3994 030264 001376 BNE 50$ ; ...
3995
3996 ; GET TRIBUTARY NUMBER
3997
3998 030266 012703 177777 60$: MOV #-1,R3 ; INDICATE NO TRIBUTARY
3999 030272 032760 000020 000000 BIT #LF,MTP,L.FLG(R0) ; MULTI-POINT CIRCUIT?
4000 030300 001002 BNE 70$ ; BR IF YES
4001 030302 000167 000546 JMP 230$ ; BR IF NO - SKIP TRIBUTARY
4002
4003 ; FIND DDCMP LINE TABLE
4004
4005 030306 012700 014610 70$: MOV #*RDCP,R0 ; GET DCP PROCESS NAME
4006 030312 CALL FNDDPV ; GET PDV INDEX
4007 030316 103464 BCS 130$ ; BR IF CAN'T FIND IT
4008 030320 016600 000012 MOV 12(SP),R0 ; RETRIEVE SLT ADDRESS
4009 030324 016602 000006 MOV 6(SP),R2 ; RETRIEVE STATION NUMBER
4010 030330 126001 000003 CMPB L,DLC(R0),R1 ; IS THIS A DDCMP LINE?
4011 030334 001055 BNE 130$ ; BR IF NO
4012 030336 032760 040000 000000 BIT #LF,RDY,L.FLG(R0) ; IS THE LINE LOADED?
4013 030344 001002 BNE 80$ ; BR IF YES - CONTINUE
4014 030346 000167 000456 JMP 210$ ; BR IF NO - DON'T DISPLAY IT
4015 030352 005067 000000G 80$: CLR $RBLCK ; ASSUME LINE TABLE IS NOT MAPPED
4016 030356 005767 000000G TST .MGMGE ; MAPPED SYSTEM?
4017 030362 100407 BMI 90$ ; BR IF NO
4018 030364 022760 120000 000010 CMP #120000,L.DLS(R0) ; IS THE LINE TABLE MAPPED?
4019 030372 101003 BHI 90$ ; BR IF NO
4020 030374 016067 000006 000000G MOV L,DLM(R0),$RBLCK ; GET LINE TABLE BIAS
4021 030402 GETAD L,DLS(R0),#L.ENA ; READ IN LINE TABLE
4022 030422 012704 000012G MOV #BFR,R4 ; POINT TO POLLING LISTHEAD
4023 030426 011405 100$: MOV (R4),R5 ; GET NEXT ENTRY
4024 030430 001417 BEQ 130$ ; BR IF END OF LIST
4025 030432 110$: GETAD R5,#S.DLCF ; READ IN TRIBUTARY STATION TABLES
4026 030450 012704 000000G MOV #BFR,R4 ; GET ADDRESS OF TRIB STATION TABLES
4027 030454 126402 000035 CMPB S,STLN(R4),R2 ; IS THIS THE TRIBUTARY?
4028 030460 001362 BNE 100$ ; BR IF NO - CONTINUE SCAN
4029 030462 116703 000034G 120$: MOVB S,STPN+$BFR,R3 ; GET TRIBUTARY NUMBER
4030 030466 000560 BR 210$ ; DISPLAY MESSAGE
4031
4032 ; FIND PCL LINE TABLE
4033
4034 030470 012700 062204 130$: MOV #*RPCL,R0 ; GET PCL PROCESS NAME
4035 030474 CALL FNDDPV ; GET PDV INDEX
4036 030500 103465 BCS 180$ ; BR IF CAN'T FIND IT
4037 030502 016600 000012 MOV 12(SP),R0 ; RETRIEVE SLT ADDRESS

```

```

4603 .SBTTL FMT CIR - FORMAT CIRCUIT-ID
4604
4605
4606
4607
4608
4609
4610
4611
4612
4613
4614
4615
4616
4617
4618
4619
4620
4621
4622
4623
4624
4625
4626
4627
4628
4629
4630
4631
4632
4633
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648
4649
4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
    034334
    034344 005003
    034346 032761 000000G 000000G
    034354 001401
    034356 005203
    034360 016101 000000G
    034364 012700 001202
    034370
    034374 124027 000040
    034400 001775
    034402 005200

    034404 112720 000055
    034410 016601 000006
    034414 116101 000013
    034420 012702 014012
    034424
    034430 005703
    034432 001412

    034434 112720 000055
    034440 016601 000006
    034444 116101 000013
    034450 012702 014012
    034454
    034460 016601 000006

    034464 032761 000020 000000
    034472 001414
    034474 105761 000014
    034500 001411
    034502 016601 000002
    034506 002406

    FMT CIR: SAVRG <R0,R1,R2,R3>
    CLR R3
    BIT #ZF.MUX,Z.FLG(R1)
    BEQ 10$
    INC R3
    10$: MOV Z,NAM(R1),R1
    MOV #TEMP3,R0
    CALL $CSTA
    20$: CMPB -(R0),#SPACE
    BEQ 20$
    INC R0

    CONTROLLER NUMBER
    MOVB #DASH,(R0)+
    MOV 6(SP),R1
    MOVB L,CTL(R1),R1
    MOV #<<3*4000>!10.>,R2
    CALL $CBTA
    TST R3
    BEQ 30$

    UNIT NUMBER
    MOVB #DASH,(R0)+
    MOV 6(SP),R1
    MOVB L,UNIT(R1),R1
    MOV #<<3*4000>!10.>,R2
    CALL $CBTA
    30$: MOV 6(SP),R1

    TRIBUTARY
    BIT #LF.MTP,L.FLG(R1)
    BEQ 40$
    TSTB L,NJIA(R1)
    BEQ 40$
    MOV 2(SP),R1
    BLT 40$

    ; SAVE REGISTERS
    ; ASSUME NOT A MUX DEVICE
    ; IS IT A MUX DEVICE?
    ; BR IF NO
    ; REMEMBER IT
    ; GET DDM PROCESS NAME
    ; STORAGE FOR CIRCUIT-ID
    ; CONVERT NAME TO ASCII
    ; DELETE TRAILING SPACES
    ;
    ; POINT PAST LAST GOOD CHARACTER

    ; INSERT DELIMITER
    ; GET SLT ADDRESS
    ; GET CONTROLLER NUMBER
    ; USE BASE 10. - FIELD WIDTH 3
    ; CONVERT TO ASCII
    ; IS THIS A MUX DEVICE?
    ; BR IF NO

    ; ELSE, INSERT DELIMITER
    ; GET SLT ADDRESS
    ; GET UNIT NUMBER
    ; USE BASE 10. - FIELD WIDTH 3
    ; CONVERT TO ASCII
    ; GET SLT ADDRESS
  
```

GF\$BUG= 000001	I.OFF = 000004	LN.DUM= 000005	LO.NET= 000002	LS.CX5= 000400
GRMS1 021536R	K\$CNT= 177546	LN.LOA= 000004	LO.NNM 000022	LS.ECH= 001000
GRMS2 021602R	K\$CSR= 177546	LN.LOO= 000003	LO.OFL 000012	LS.FDX= 004000
GRMS3 021660R	K\$LDC= 000000	LN.OAU= 000003	LO.ONM 000004	LS.HDX= 010000
GRPBIL 000750R	K\$TPS= 000074	LN.OFF= 000001	LO.OUT 000056	LS.HLP= 000012
GTEKEC 034542R	K.CSR 000002	LN.ON = 000000	LO.OWN 000026	LS.LIN= 000003
GTSTID 034630R	K.PRI 000000	LN.OOP= 000004	LO.PAR 000010	LS.LMC= 000007
G\$CUG 000010	K.VCT 000001	LN.OPE= 000001	LO.PRI 000012	LS.LOG= 000020
G\$DTE 000012	LACTIV 000441R	LN.REF= 000002	LO.RET 000064	LS.MOD= 000036
G\$FLG 000014	LCNTRL 000567R	LN.SER= 000002	LO.RPA 000030	LS.NOB= 000014
G\$GNAM 000016	LD\$LP = 000000	LN.STA= 000017	LO.SCR 000016	LS.NTI= 000013
G\$LEN 000018	LD.ALI= 000030	LN.SUB= 000360	LO.SEG 000054	LS.OBJ= 000022
G\$LNK 000000	LD.CEX= 000024	LN.TRI= 000006	LO.SKA 000014	LS.OPT= 000400
G\$NAM 000002	LD.CIR= 000035	LOACTV 000236R	LO.SKN 000014	LS.PRO= 000002
G\$NML 000015	LD.LIN= 000025	LOCON 000340R	LO.SNM 000035	LS.TOP= 002000
G\$STP= 000000	LD.LOG= 000031	LOERR1 004552R	LO.TPA 000020	LS.TST= 000010
G\$STSS= 000000	LD.MOD= 000037	LOERR2 004576R	LO.TRB 000034	LS.UNF= 020000
G\$STTK= 000000	LD.NOD= 000027	LOFIL 000333R	LO.TRI 000030	LS.XIT= 000011
G\$WRD= 000000	LD.OBJ= 000032	LOGIN 000120R	LO.UNT 000033	LTRIB 000621R
HEXASC 035546R	LD.PRO= 000026	LOGTYP 001676R	LO.VCT 000014	LX.ALI= 000017
HF\$DLM= 000002	LE.NRH= 000200	LOKNWN 000275R	LO.VER 000042	LX.CEX= 000004
HF\$GWY= 000010	LE.NRO= 000001	LOMON 000323R	LPDINT 000605R	LX.CIR= 000034
HF\$HDS= 000004	LE.NSH= 000002	LMSG1 004102R	LP\$EIP= 002000	LX.LIN= 000006
HF\$XDF= 000020	LE.NSO= 000020	LMSG2 004126R	LP\$ENB= 004000	LS.LOG= 000021
H\$CUG 000010	LE.RCO= 000040	LMSG3 004174R	LP\$PCT= 001400	LX.NOB= 000015
H\$DST 000012	LE.SAF= 000004	LMSG4 004432R	LP\$TMR= 100000	LS.OBJ= 000023
H\$D29 000014	LE.STT= 000010	LMSG5 004440R	LP\$UP = 010000	LX.PRO= 000005
H\$FLG 000000	LE.XTU= 000100	LMSG6 004464R	LP\$WTD= 020000	LX25 000641R
H\$GLEN 000104	LF.ACT= 100000	LMSG7 004512R	LP\$WTS= 040000	L\$ACHN 000012
H\$GLT 000044	LF.BRO= 000400	LMSG8A 004412R	LR.ACC 000020	L\$APVC 000014
H\$GNAM 000050	LF.BW= 000007	LMSG3X 004256R	LR.ACO 000052	L\$ASVC 000010
H\$GNML= 000020	LF.ENA= 002000	LMSG3Y 004332R	LR.ADD 000012	L\$AUC 000042
H\$GPT 000046	LF.LPB= 001000	LONGEV 000350R	LR.ALI 000002	L\$CHLS 000034
H\$HITS 000034	LF.MDC= 000100	LOSPEC 000313R	LR.BLK 000010	L\$CHTB 000030
H\$HLEN 000044	LF.MFL= 004000	LO.ADD 000022	LR.CTL 000004	L\$CLEN= 000044
H\$LBDA 000070	LF.MTP= 000020	LO.ADR 000052	LR.DES 000010	L\$CTEN 000032
H\$LBDN 000072	LF.PAC= 000200	LO.CLS 000002	LR.HLP 000002	L\$CTIM 000040
H\$LDTE 000002	LF.RDY= 040000	LO.CON 000006	LR.LIN 000002	L\$DTEA 000020
H\$LEN 000042	LF.REA= 010000	LO.COP 000010	LR.NAM 000012	L\$DTEL 000017
H\$LOTS 000032	LF.SER= 000040	LO.COS 000020	LR.PAS 000041	L\$GLEN 000134
H\$NETW 000024	LF.TIM= 000010	LO.CSR 000010	LR.PRO 000002	L\$LEN 000122
H\$NML = 000006	LF.UNL= 020000	LO.CTL 000032	LR.STS 000000	L\$LLCH 000016
H\$NPT 000022	LF.X2P= 000000	LO.DES 000002	LR.TRI 000006	L\$LNK 000000
H\$PTB 000020	LIERR1 015536R	LO.DTE 000047	LR.TYP 000002	L\$MCHN 000036
H\$PVC 000006	LKNOWN 000456R	LO.EVT 000004	LR.UCB 000016	L\$NETW 000114
H\$RDTE 000004	LLAB 000645R	LO.GRO 000024	LR.UIC 000020	L\$NIRE 000072
H\$RNW 000042	LMSG1 033732R	LO.HOS 000040	LR.UNT 000005	L\$NIRE 000104
H\$SVC 000036	LMSG12 034164R	LO.HTM 000032	LSPEC 000472R	L\$NMAC 000076
H\$TRB 000016	LMSG2 033756R	LO.INA 000062	LS\$GDN= 000004	L\$NMAS 000074
H\$XAVL 000100	LMSG3 034014R	LO.INC 000060	LS\$OFF= 000000	L\$NMST 000003
H\$XBIA 000074	LMSG4 034106R	LO.LDT 000046	LS\$STR= 000002	L\$NNRE 000106
H\$X29C 000040	LMSG6 034134R	LO.LID 000030	LS\$UP = 000003	L\$NRBY 000046
INSPCT 000067R	LMSG9 034156R	LO.LIN 000004	LS\$WT = 000001	L\$NRCA 000066
IO.ATT= ***** GX	LN\$UFF= 000001	LO.LTM 000034	LS.ALI= 000016	L\$NRFS 000100
IO.DET= ***** GX	LN\$ON = 000000	LO.MAC 000022	LS.CEX= 000001	L\$NRPK 000056
IS\$RAR= 000070	LN\$SHU= 000002	LO.MDE 000024	LS.CIR= 000033	L\$NRRE 000105
IS\$RDV= 000000	LN.CLO= 000000	LO.NAM 000044	LS.CXO= 010000	L\$NRST 000107

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 8 C 9
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
LO.LID	000030	#5-99
LO.LIN	000004	#5-99
LO.LTM	000034	#5-99
LO.MAC	000022	#5-99
LO.MDE	000024	#5-99
LO.NAM	000044	#5-99
LO.NET	= 000002	#5-99 33-2435
LO.NNM	000022	#5-99
LO.OFL	000012	#5-99
LO.ONM	000004	#5-99
LO.OUT	000056	#5-99
LO.OWN	000026	#5-99
LO.PAR	000010	#5-99
LO.PRI	000012	#5-99
LO.RET	000064	#5-99
LO.RPA	000030	#5-99
LO.SCR	000016	#5-99
LO.SEG	000054	#5-99
LO.SKA	000014	#5-99 *12-681 17-981 17-984
LO.SKN	000014	#5-99 12-675
LO.SNM	000035	#5-99
LO.TPA	000020	#5-99
LO.TRB	000034	#5-99
LO.TRI	000030	#5-99
LO.UNT	000033	#5-99
LO.VCT	000014	#5-99
LO.VER	000042	#5-99
LPOINT	000605 P	#6-215 53-3992 55-4432
LR.ACC	000020	#5-99
LR.ACO	000052	#5-99
LR.ADD	000012	#5-99 26-1537 26-1552 26-1554
LR.ALI	000002	#5-99 10-496
LR.BLK	000010	#5-99
LR.CTL	000004	#5-99 27-2028 28-2201
LR.DES	000010	#5-99 10-508
LR.HLP	000002	#5-99
LR.LIN	000002	#5-99 5-99 5-99 5-99 5-99 5-99 5-99 27-2004 27-2021
LR.NAM	000012	#5-99 28-2194
LR.PAS	000041	#5-99 26-1529 26-1531 26-1533 26-1545 26-1547 26-1549
LR.PRO	000002	#5-99 11-606
LR.STS	000000	#5-99 39-2883 59-4737
LR.TRI	000006	#5-99 27-2043
LR.TYP	000002	#5-99 9-416
LR.UCB	000016	#5-99 10-508
LR.UIC	000020	#5-99
LR.UNT	000005	#5-99 27-2035 28-2208
LSPEC	000472 R	#6-197 21-1219 28-2186
LS.ALI	= 000016	#5-99
LS.CEX	= 000001	#5-99
LS.CIR	= 000033	#5-99
LS.CXO	= 010000	#5-99 59-4737

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 21 C 10
 SYMBOL CROSS REFERENCE CREF 04.00
 SYMBOL VALUE REFERENCES

#26-1894	26-1896	26-1896	#26-1896	#26-1897	26-1899	26-1899	#26-1899	#26-1900
26-1902	26-1902	#26-1902	#26-1903	27-2136	27-2136	#27-2136	#27-2137	28-2240
28-2240	#28-2241	29-2280	29-2280	#29-2280	#29-2281	31-2375	31-2375	31-2375
#31-2375	#31-2376	31-2378	31-2378	#31-2378	#31-2379	31-2381	31-2381	#31-2381
#31-2382	31-2384	31-2384	#31-2384	#31-2385	33-2492	33-2492	#33-2492	#33-2493
33-2495	33-2495	#33-2495	#33-2496	33-2498	33-2498	#33-2498	#33-2499	33-2501
33-2501	#33-2501	#33-2502	33-2504	33-2504	#33-2504	#33-2505	33-2507	33-2507
#33-2507	#33-2508	33-2510	33-2510	#33-2510	#33-2511	34-2602	34-2602	#34-2602
#34-2603	34-2607	34-2607	#34-2607	#34-2608	34-2610	34-2610	#34-2610	#34-2611
34-2613	34-2613	#34-2613	#34-2614	34-2616	34-2616	#34-2616	#34-2617	35-2678
35-2678	#35-2678	#35-2679	35-2681	35-2681	#35-2681	#35-2682	35-2684	35-2684
#35-2684	35-2685	37-2787	37-2787	#37-2787	#37-2788	37-2790	37-2790	#37-2790
#37-2791	37-2793	37-2793	#37-2793	39-2901	39-2901	#39-2901	#39-2901	#39-2902
39-2906	39-2906	#39-2906	#39-2907	40-3070	40-3070	#40-3070	40-3071	40-3073
40-3073	#40-3073	#40-3074	40-3076	40-3076	#40-3076	#40-3077	40-3079	40-3079
#40-3079	40-3080	40-3082	40-3082	40-3082	#40-3083	40-3085	40-3085	40-3085
#40-3086	40-3088	40-3088	#40-3088	40-3089	40-3091	40-3091	#40-3091	#40-3092
40-3094	40-3094	#40-3094	40-3095	40-3097	40-3097	#40-3097	40-3098	40-3100
40-3100	#40-3100	#40-3101	40-3103	40-3103	#40-3103	#40-3104	40-3106	40-3106
#40-3106	#40-3107	40-3109	40-3109	#40-3109	#40-3110	42-3189	42-3189	#42-3189
#42-3190	42-3194	42-3194	#42-3194	42-3195	48-3512	48-3512	#48-3512	#48-3513
48-3515	48-3515	#48-3515	48-3516	48-3518	48-3518	#48-3518	48-3519	48-3521
48-3521	#48-3521	#48-3522	48-3525	48-3525	#48-3525	48-3526	48-3528	48-3528
#48-3528	48-3529	49-3657	49-3657	#49-3657	#49-3658	49-3660	49-3660	#49-3660
#49-3661	49-3663	49-3663	#49-3663	49-3664	49-3666	49-3666	#49-3666	#49-3667
50-3723	50-3723	#50-3723	#50-3724	50-3726	50-3726	#50-3726	50-3727	50-3729
50-3729	#50-3729	#50-3730	51-3824	51-3824	#51-3824	51-3825	51-3827	51-3827
#51-3827	#51-3828	51-3830	51-3830	#51-3830	51-3831	51-3833	51-3833	#51-3833
#51-3834	51-3836	51-3836	#51-3836	#51-3837	51-3839	51-3839	51-3839	51-3840
51-3842	51-3842	#51-3842	51-3843	51-3845	51-3845	#51-3845	51-3846	51-3848
51-3848	#51-3848	#51-3849	51-3851	51-3851	#51-3851	#51-3852	52-3916	52-3916
#52-3916	#52-3917	52-3919	52-3919	#52-3919	#52-3920	53-4283	53-4283	#53-4283
#53-4284	53-4286	53-4286	#53-4286	53-4287	53-4289	53-4289	#53-4289	#53-4290
53-4292	53-4292	#53-4292	53-4293	53-4295	53-4295	#53-4295	53-4296	53-4298
53-4298	#53-4298	#53-4299	53-4301	53-4301	#53-4301	#53-4302	53-4304	53-4304
#53-4304	53-4305	53-4308	53-4308	#53-4308	53-4309	53-4311	53-4311	#53-4311
#53-4312	54-4362	54-4362	#54-4362	54-4363	54-4365	54-4365	#54-4365	#54-4366
54-4368	54-4368	#54-4368	54-4369	54-4371	54-4371	#54-4371	54-4372	55-4529
55-4529	#55-4529	#55-4530	55-4532	55-4532	#55-4532	#55-4533	55-4535	55-4535
#55-4535	#55-4536	55-4538	55-4538	#55-4538	55-4541	55-4541	#55-4541	#55-4541
#55-4542	55-4544	55-4544	#55-4544	#55-4545	55-4547	55-4547	#55-4547	#55-4548
#6-236	6-237	#8-351	8-353	#8-355	8-357	#8-362	8-363	#9-434
9-435	#10-547	10-548	10-550	10-551	#11-640	11-641	12-760	12-761
#12-763	12-764	#12-766	12-767	#12-769	12-770	#12-772	12-773	#12-776
12-777	#12-779	12-780	#12-782	12-783	#12-785	12-786	#12-788	12-789
#12-791	12-792	#12-794	12-795	#26-1830	26-1831	#26-1833	26-1834	#26-1836
26-1837	#26-1839	26-1840	#26-1842	26-1843	26-1845	#26-1846	26-1848	26-1849
#26-1851	26-1853	#26-1855	26-1857	#26-1859	26-1860	#26-1862	26-1863	#26-1865
26-1866	#26-1868	26-1869	#26-1871	26-1872	#26-1874	26-1875	26-1877	6-1878
#26-1880	26-1881	#26-1884	26-1887	#26-1889	26-1891	#26-1893	26-1894	#26-1896
26-1897	#26-1899	26-1900	#26-1902	26-1903	#27-2136	27-2137	#28-2240	28-2241
#29-2280	29-2281	#31-2375	31-2376	#31-2378	31-2379	#31-2381	31-2382	#31-2384

\$\$\$ = 034162 R

```

171
172
173
174
175
176
177
178
179
180
181 000240 010000 012144'
182 000244 000004 012536'
183 000250 000010 013746'
184 000254 000020 014070'
185 000260 000000
186
187
188
189
190
191
192
193
194 000262 000002 014142'
195 000266 010000 014240'
196 000272 000004 014666'
197 000276 000000

      .SBTTL DISPATCH TABLES

      ;
      ; PARAMETER VALIDITY DISPATCH TABLE - THIS TABLE CONTROLS THE DISPATCHING
      ; OF THE PROPER ROUTINE TO CHECK FOR THE VALIDITY OF AN INPUT
      ; PARAMETER FOR A SET OR CLEAR LOGGING COMMAND
      ;
      ; EACH ENTRY HAS THE FOLLOWING FORMAT:
      ; WORD 1: BIT TO INDICATE PARAMETER IS PRESENT (0 = END OF TABLE)
      ; WORD 2: ADDRESS OF VALIDITY CHECKING ROUTINE
      ;
      PVALID: .WORD OP$EVE,CKEVE      ; EVENTS PARAMETER
      .WORD OP$NAM,CKNAME      ; SINK NAME PARAMETER
      .WORD OP$SIN,CKSINK      ; SINK NODE PARAMETER
      .WORD OP$NOD,CKNODE      ; NODE ENTITY PARAMETER
      .WORD 0                  ; END OF TABLE

      ;
      ; PARAMETER CHANGE DISPATCH TABLE - THIS TABLE CONTROLS THE DISPATCHING
      ; OF THE PROPER ROUTINE TO CHANGE A LOGGING PARAMETER
      ;
      ; EACH ENTRY HAS THE FOLLOWING FORMAT:
      ; WORD 1: BIT TO INDICATE PARAMETER IS PRESENT (0 = END OF TABLE)
      ; WORD 2: ADDRESS OF ROUTINE TO CHANGE PARAMETER
      ;
      CHTBLE: .WORD OP$LST,CHSTA      ; CHANGE LOGGING STATE
      .WORD OP$EVE,CHFIL      ; CHANGE FILTER MASKS
      .WORD OP$NAM,CHNAM      ; CHANGE LOGGING SINK NAME
      .WORD 0                  ; END OF TABLE

```

C 12

```

681 .SBTTL $VOWN - SET LINE OWNER
682
683 *--$VOWN - SET LINE OWNER
684
685 INPUTS:
686 $SLTA = SLT ADDRESS
687 RQB = REQUEST BLOCK
688
689 OUTPUTS:
690 C-BIT = SUCCESS/FAILURE
691 ALL REGISTERS DESTROYED
692
693
694
695 $VOWN::
696     MOV     RQB+LO.OWN,R0      ; GET THE PDV PROCESS NAME
697     CALL    FNPID             ; TRY TO FIND THE PDV ID
698     BCS     101$              ; IF CS, NOT FOUND
699     TST     R0                ; IS THIS FOR AUX ?
700     BEQ     101$              ; IF EQ, YES .. ERROR
701     CALL    FINDFR            ; FIND A FREE CHANNEL
702     BCS     104$              ; IF CS, NO FREE CHANNELS
703     MOV     $SLTA,R1          ; ELSE, GET THE SLT ADDRESS
704     BIT     #LF.RDY,L.FLG(R1) ; IS THE LINE READY ?
705     BEQ     102$              ; IF EQ, NO .. ERROR
706     BIT     #LF.ACT,L.FLG(R1) ; ELSE, IS THE LINE ACTIVE ?
707     BNE     102$              ; IF NE, YES .. ERROR
708     MOV     L,NSTA(R1),R2     ; ELSE, GET THE NUMBER OF TRIBUTARIES
709     BNE     10$               ; IF NE, LINE IS MULTIPOINT
710
711 ; PT-TO-PT LINE
712
713     BIT     #LF.LPB,L.FLG(R1) ; IS THIS LINE IN LOOPBACK ?
714     BNE     102$              ; IF NE YES
715     MOV     $SLN,(R3)         ; STORE SYSTEM LINE NUMBER IN LLC TABLE
716     MOV     R0,L.OWNR(R1)    ; SET NEW LINE OWNER
717     MOV     $SLN,R1          ; GET SLN
718     ASL     R1                ; MAKE IT A WORD OFFSET
719     ADD     $LLCTA,R1         ; POINT AT ENTRY IN REV. MAPPING TABLE
720     TST     (R1)              ; IS THIS POINTING TO A STATION TABLE ?
721     BMI     105$              ; IF MI, YES .. SOMETHING'S WRONG
722     BR      20$               ; AND LEAVE
723
724 ; MULTIPOINT LINE
725
726     10$: CMP     R2,$TRIB      ; IS THE TRIBUTARY ON THIS LINE ?
727     BLOS    105$              ; IF LOS, NO .. ERROR
728     MOV     $TRIB,R2          ; ELSE, GET TRIBUTARY NUMBER
729     ASL     R2                ; CONVERT TO A DOUBLE WORD INDEX
730     ASL     R2
731     ADD     R1,R2             ; POINT INTO SLT
732     ADD     #L.MPF,R2         ; POINT TO STATION INFO IN SLT
733     BIT     #SF.LPB,S.FLG(R2) ; IS THIS STATION IN LOOPBACK ?
734     BNE     102$              ; IF NE, YES
735     MOV     $SLN,(R3)         ; ELSE, STORE SLN AND STATION NUMBER
736     MOV     $TRIB,1(R3)       ; ...
737     MOV     R0,S.OWNR(R2)     ; AND SET NEW OWNER

```

D 12


```

1213 003620          50$: RETURN
1214
1215          ; ERROR CONDITIONS
1216
1217 003622          101$: ERRPT$ OPT1,$EROUT
1218 003632          ERMMSG$ <Invalid node option>
1219 003655          ERBLK$ OPT1
1220
1221          .DSABL LSB
1222
1223          ; SET NODE REQUESTS
1224
1225          SNODTB:
1226 003660 020662'    .WORD $VHST          ; SET HOST
1227 003662 004234'    .WORD $VLIN          ; SET NODE LINE
1228 003664 004654'    .WORD $VREM          ; SET NODE NAME
1229 003666 006130'    .WORD $RPASW        ; SET RECEIVE PASSWORD
1230 003670 006260'    .WORD $TPASW        ; SET TRANSMIT PASSWORD
1231 003672 006410'    .WORD $VERIF        ; SET VERIFICATION STATE
1232 003674 006524'    .WORD $VACP         ; SET EXECUTOR STATE
1233 003676 177777'    .WORD -1            ; NOT USED (NEEDED FOR POSITION)
1234 003700 004020'    .WORD $VSEG         ; SET EXECUTOR SEG BUF SIZE
1235 003702 004054'    .WORD $VINC         ; SET EXEC INCOMING TIMER
1236 003704 004110'    .WORD $VOUT        ; SET EXEC OUTGOING TIMER
1237 003706 004144'    .WORD $VINA        ; SET EXEC INACTIVITY TIMER
1238 003710 004200'    .WORD $VRET        ; SET EXEC RETRANSMIT FACTOR
1239 003712 000000'    .WORD 0
1240
1241          ; CLEAR NODE REQUESTS
1242
1243          CNODTB:
1244 003714 020662'    .WORD $VHST          ; SET HOST
1245 003716 004234'    .WORD $VLIN          ; SET NODE LINE
1246 003720 004654'    .WORD $VREM          ; SET NODE NAME
1247 003722 006130'    .WORD $RPASW        ; SET RECEIVE PASSWORD
1248 003724 006260'    .WORD $TPASW        ; SET TRANSMIT PASSWORD
1249 003726 000000'    .WORD 0

```

```

1752          .SBTTL  SETBIT - SET LOOPBACK BIT
1753      ;+
1754      ;**--SETBIT-SET LOOPBACK BIT
1755      ;
1756      ; INPUTS:
1757      ;     R2 = CHANNEL NO.
1758      ;
1759      ; OUTPUTS:
1760      ;     C-BIT = SUCCESS/FAILURE
1761      ;     R2,R3 = DESTROYED
1762      ;
1763      ; -
1764
1765 005774      SETBIT:
1766 005774      CALL    GTSLT          ; GET SLT ADDRESS
1767 006000 103420      BCS     20$          ; IF CS, NO XPT
1768 006002 105762      TSTB    L,NSTA(R2)  ; IS THIS LINE MULTIPOINT ?
1769 006006 001004      BNE     10$          ; IF NE, YES
1770 006010 052762      000000G 000000G    BIS     #LF.LPB,L.FLG(R2) ; ELSE, SET THE LOOP BACK BIT
1771 006016 000411      BR      20$          ; AND RETURN
1772 006020 006303      10$:  ASL     R3          ; POINT TO STATION INFO IN SLT
1773 006022 006303      ASL     R3          ; ...
1774 006024 060302      ADD     R3,R2        ; ...
1775 006026 060702      000000G      ADD     #L.MPF,R2 ; ...
1776 006032 152762      000000G 000000G    BISB   #SF.LPB,S.FLG(R2) ; AND SET THE LOOPBACK BIT
1777 006040 000241      CLC          ; INDICATE SUCCESS
1778 006042      20$:  RETURN

```

```

2344 011202          SWITCHO  #HMBUF          ; USE TEMPORARY BUFFER
2345 011210          PUTRC    $DECT,#D$END    ; WRITE BACK UPDATED HOME BLOCK
2346 011230          SWITCHO          ; SWITCH TO DEFAULT OUTPUT BUFFER
2347
2348
2349
2350 011236 000404          BR      15$          ; AND BR TO DEALLOCATE IT
2351
2352 011240 010346          10$: MOV    R3,-(SP)          ; GET UNMAPPED ADDR OF PREVIOUS BLOCK
2353 011242 010046          MOV    R0,-(SP)          ; GET UNMAPPED ADDRESS OF CURRENT BLOCK
2354 011244          CALL    $XLINK          ; UNLINK BLOCK FROM LIST
2355 011250 012601          15$: MOV    (SP)+,R1          ; RETRIEVE LENGTH OF ALIAS BLOCK
2356 011252          CALL    $DEPOL          ; DEALLOCATE THE ALIAS BLOCK
2357 011256 000410          BR      22$          ; AND RETURN
2358
2359          ; SET REQUEST
2360
2361 011260 032767 000001 000000G 20$: BIT    #OP$SCO,$O$PTON          ; WAS THE SCOPE GIVEN ?
2362 011266 001005          BNE     23$          ; IF NE, YES .. OKAY
2363 011270          ERRPT$    ERR26,$EROUT          ; ELSE, SCOPE PARAMETER MISSING
2364 011300 000572          BR      22$          ; CONTINUE
2365 011302 032767 000000G 000000G 23$: BIT    #VF.ADD,$V+IAG          ; WAS THE DESTINATION GIVEN AS ADDRESS
2366 011310 001415          BEQ     25$          ; IF EQ, NO .. OKAY
2367 011312 016701 000010G          MOV    RQB+LR.DES,R1          ; ELSE, GET THE NODE ADDRESS
2368 011316          CALL    $FNADD          ; AND FIND THE REMOTE NODE BLOCK
2369 011322 103572          BCS     103$          ; IF CS, NODE NOT IN SYSTEM
2370 011324 012701 000010G          MOV    #RQB+LR.DES,R1          ; GET OUTPUT BUFFER
2371 011330 012702 000002G          MOV    #BFR+R.NAM,R2          ; AND GET INPUT BUFFER
2372 011334 012703 000006          MOV    #6,R3          ; AND LENGTH TO MOVE
2373 011340          CALL    MOVE          ; STORE NODE NAME IN REQUEST BLOCK
2374 011344          CALL    $FNALI          ; TRY TO FIND THE ALIAS BLOCK
2375 011350 103402          BCS     30$          ; IF CS, NOT FOUND .. OKAY
2376 011352          CALL    REMALI          ; ELSE, REMOVE THE ALIAS
2377 011356 012701 000023          MOV    #A.ACC+3,R1          ; CALCULATE LENGTH OF ALIAS BLOCK
2378 011362 116700 000020G          MOVB   RQB+LR.UIC,R0          ; GET SIZE OF FIELD
2379 011366 060001          ADD     R0,R1          ; SKIP TO NEXT FIELD
2380 011370 116700 000041G          MOVB   RQB+LR.PAS,R0          ; GET SIZE OF FIELD
2381 011374 060001          ADD     R0,R1          ; ADD IN SIZE
2382 011376 116700 000052G          MOVB   RQB+LR.ACO,R0          ; GET SIZE OF FIELD
2383 011402 060001          ADD     R0,R1          ; ADD IN SIZE OF FIELD
2384 011404          CALL    $ALPOL          ; TRY TO ALLOCATE THE BUFFER
2385 011410 103533          BCS     102$          ; IF CS, ALLOCATION FAILURE
2386 011412 010067 000164'          MOV    R0,CURUNM          ; SAVE UNMAPPED ADDRESS
2387 011416 010146          MOV    R1,-(SP)          ; SAVE SIZE OF ALIAS
2388
2389          SWITCHI  #HMBUF          ; USE TEMPORARY BUFFER
2390 011420          GETRV    $DECT,#D$END    ; READ DECNET HOME BLOCK
2391 011426          SWITCHI          ; RESET DEFAULT BUFFER
2392
2393 011454 012701 000000G          MOV    #BFR,R1          ; GET ADDRESS OF OUTPUT BUFFER
2394 011460 005021          CLR     (R1)+          ; SKIP LINK POINTER
2395 011462 012702 000002G          MOV    #RQB+LR.ALI,R2          ; GET ADDRESS OF INPUT BUFFER
2396 011466 012703 000006          MOV    #6,R3          ; GET LENGTH OF ALIAS FIELD
2397 011472          CALL    MOVE          ; STORE ALIAS NAME
2398 011476 016721 000016G          MOV    RQB+LR.UCB,(R1)+          ; STORE UCB ADDRESS
2399 011502 012702 000010G          MOV    #RQB+LR.DES,R2          ; GET ADDRESS OF INPUT BUFFER
2400 011506 012703 000006          MOV    #6,R3          ; GET LENGTH OF DESTINATION FIELD

```

CHKFIL - CHECK FILE SINK NAME

```

2898 013504 022702 000073      CMP    #' ,R2      ; VALID TERMINATOR
2899 013510 001025      BNE    120$      ; BR IF NO - ERROR
2900 013512 010167 000142'    MOV    R1,FILTYP  ; STORE FILE TYPE
2901 013516 000407      BR      100$      ; ADN CHECK FOR VERSION NUMBER
2902 013520 010167 000142'    MOV    R1,FILTYP  ; SAVE FILE TYPE
2903 013524 020500      CMP    R5,R0      ; END OF STRING?
2904 013526 101414      BLOS    110$      ; BR IF YES
2905 013530 122027 000073    CMPB   (R0)+, #' ; VALID TERMINATOR?
2906 013534 001013      BNF     120$      ; BR IF NO
2907
2908      ; CHECK VERSION NUMBER
2909
2910 013536 020500      100$:  CMP    R5,R0      ; END OF STRING?
2911 013540 101407      BLOS    110$      ; BR IF YES
2912 013542      CALL    $COTB      ; CONVERT NUMBER TO OCTAL
2913 013546 022701 077777      CMP    #MXVER,R1  ; LEGAL VERSION NUMBER?
2914 013552 103404      BLO     120$      ; BR IF NO
2915 013554 010167 00014'    MOV    R1,FILVER  ; SAVE VERSION NUMBER
2916 013560 000241      110$:  CLC      ; INDICATE SUCCESS
2917 013562 000401      BR      130$      ; AND EXIT
2918 013564 000261      120$:  SEC      ; INDICATE FAILURE
2919 013566 012605      130$:  MOV    (SP)+,R5  ; RESTORE R5
2920 013570      RETURN

```

```
.SBTTL  BLDLIN - BUILD LINE ENTITY
```

E 1

```

1815 011774          ERRPT$  NMSG15,MSGOUT          ; AND DISPLAY
1816          :
1817          : AND FINALLY DISPLAY 'DLX TYPE'
1818          :
1819 012004          ERRPT$  NMSG16,MSGOUT          ; DLX TYPE
1820          :
1821 012014          510$:  ERRPT$  NMSG4,MSGOUT          ; MAKE OUTPUT LOUK NICE
1822          :
1823 012024          520$:  DIR$   #TIDET          ; DETACH TERMINAL
1824 012032          RETURN
1825          :
1826          .ENABL  LC
1827          :
1828          : DISPLAY MESSAGES
1829          :
1830 012034          ERMSG$ <%N%A %A as of %Y>
1831 012054          ERBLK$  NMSG1,<TEMP2,TEMP1>
1832          :
1833 012064          ERMSG$ <%N%Executor node = %D (%A)%N>
1834 012117          ERBLK$  NMSG2,<TEMP1,TEMP2>
1835          :
1836 012126          ERMSG$ <%N%Executor node = %D.%D (%A)%N>
1837 012165          ERBLK$  NMSG2A,<TEMP7,TEMP1,TEMP2>
1838          :
1839 012176          ERMSG$ < %State = %A, Identification = %A>
1840 012240          ERBLK$  NMSG3,<TEMP1,TEMP2>
1841          :
1842 012250          ERMSG$ < >
1843 012251          ERBLK$  NMSG4
1844          :
1845 012254          ERMSG$ < Host = %D (%A), Maximum links = %D>
1846 012321          ERBLK$  NMSG5,<TEMP3,TEMP2,TEMP1>
1847          :
1848 012332          ERMSG$ < Host = %D.%D (%A), Maximum links = %D>
1849 012402          ERBLK$  NMSG5A,<TEMP7,TEMP3,TEMP2,TEMP1>
1850          :
1851 012416          ERMSG$ < Maximum address = %D, Maximum circuits = %B,>
1852 012475          ERMSG$ < Maximum cost = %D>
1853 012517          ERBLK$  NMSG6,<TEMP4,TEMP1,TEMP3>
1854          :
1855 012530          ERMSG$ < Maximum area = %D, Maximum area cost = %D>
1856 012604          ERMSG$ < Maximum area hops = %D>
1857 012635          ERBLK$  NMSG6A,<TEMP1,TEMP2,TEMP3>
1858          :
1859 012646          ERMSG$ <% Router adjacencies = %D, End.ode adjacencies = %D>
1860 012732          ERBLK$  NMSG6B,<TEMP4,TEMP7>
1861          :
1862 012742          ERMSG$ < Maximum hops = %D, Maximum visits = %D>
1863 013013          ERBLK$  NMSG7,<TEMP4,TEMP3>
1864          :
1865 013022          ERMSG$ < Type = %A>
1866 013036          ERBLK$  NMSG8,<TEMP8>
1867          :
1868 013044          ERMSG$ < Receive password = ...>
1869 013075          ERBLK$  NMSG9
1870          :
1871 013100          ERMSG$ < Transmit password = ...>

```

```

2387 .SBTTL NXTRDT - Get next RDT block ;[TD001]
2388 ;[TD001]
2389 ;[TD001]
2390 *** NXTRDT - Find next RDT block to operate on ;[TD001]
2391 ;[TD001]
2392 INPUTS: ;[TD001]
2393 $BFR - Address of next RDT descriptor ;[TD001]
2394 $OPTON - Options word ;[TD001]
2395 LO,DES+RQB - Destination to look for ;[TD001]
2396 ;[TD001]
2397 OUTPUTS: ;[TD001]
2398 Carry clear: ;[TD001]
2399 $BFR contains next RDT block ;[TD001]
2400 Carry set: ;[TD001]
2401 End of RDT List ;[TD001]
2402 ;[TD001]
2403 NXTRDT: CLR R0 ; Assume no match required ;[TD001]
2404 BIT #OP$DST,$OPTON ; Check if specific destination set ;[TD001]
2405 BEQ 10$ ; Branch if none ;[TD001]
2406 MOV #RQB+LO,DES,R0 ; Get address of destination to match ;[TD001]
2407 10$: MOV #R$LEN,R1 ; Length of RDT block ;[TD001]
2408 MOV #R$NAM,R2 ; Offset of DTE name ;[TD001]
2409 MOV #-6,R3 ; Negated length of DTE name ;[TD001]
2410 CALL FNDBLK ; Find matching block ;[TD001]
2411 RETURN ;[TD001]

016522 005000
016524 03277 000200 000000G
016532 001402
016534 012700 000002G
016540 012701 000024
016544 012702 000002
016550 012703 177772
016554
016560

```

```

2909 .SBTTL DMDST - SHOW MODULE X2n-SERVER DESTINATION ;[TD001]
2910 ;**~1 ;[TD001]
2911
2912 *** DMDST - SHOW MODULE X2n-SERVER DESTINATION ;[TD001]
2913 ;[TD001]
2914 INPUTS: ;[TD001]
2915 R0 - Index into home block to appropriate listhead ;[TD001]
2916 ;[TD001]
2917 OUTPUTS: ;[TD001]
2918 Destination info displayed on user's terminal ;[TD001]
2919 ;[TD001]
2920 NOTE ;[TD001]
2921 This entire routine is dependent on the ordering of fields ;[TD001]
2922 in the destination block. ;[TD001]
2923 ;[TD001]
2924
2925 DMDST: ADD $PSIPT,R0 ; Point to listhead ;[TD001]
2926 GETRV R0,#2 ; Read in destination listhead ;[TD001]
2927 10$: CALL NXTDST ; Get next destination in list ;[TD001]
2928 BCC 11$ ;[TD001]
2929 JMP 90$ ; Escape if end of list ;[TD001]
2930 11$: MOV #TEMP1,R2 ; Output buffer for destination name ;[TD001]
2931 TST R1 ; Check destination name length ;[TD001]
2932 DEQ 15$ ;[TD001]
2933 12$: MOVB (R0)+(R2)+ ; Move destination to output ;[TD001]
2934 SOB R1,12$ ;[TD001]
2935 15$: CLRB (R2) ; Create ASCII string ;[TD001]
2936 ERRT$ DSMS1,MSGOUT ; Display destination name ;[TD001]
2937 ;[TD001]
2938 022426 032767 000000G 000000G BIT #PF.XDF,$PFLAG ; Check for extended data format ;[TD001]
2939 022434 001535 BEQ 50$ ; Branch if old-format ;[TD001]
2940 ;[TD001]
2941 ; Display node, accounting data, priority and object for extended block ;[TD001]
2942 ;[TD001]
2943 022436 122767 000040 000010G CMPB #SPACE,$BFR+D$NOD ; Check for destination node ;[TD001]
2944 022444 001416 BEQ 34$ ; Branch if not present ;[TD001]
2945 022446 012700 MOV #BFR+D$NOD,R0 ; Address of node name ;[TD001]
2946 022452 012702 MOV #TEMP1,R2 ; Address to receive node name ;[TD001]
2947 022456 012701 000006 MOV #6,R1 ; Length of node name ;[TD001]
2948 022462 112022 32$: MOVB (R0)+(R2)+ ; Move node name ;[TD001]
2949 022464 SOB R1,32$ ;[TD001]
2950 022470 105012 CLRB (R2) ; Make ASCII string ;[TD001]
2951 022472 ERRT$ DSMS9,MSGOUT ; Display node ;[TD001]
2952 022502 012700 000032G 34$: MOV #BFR+D$GVL,R0 ; Address start of variable data ;[TD001]
2953 022506 116701 000022G MOVB $BFR+D$USL,R1 ; Skip destination field ;[TD001]
2954 022512 060100 ADD R1,R0 ;[TD001]
2955 022514 116701 000023G MOVB $BFR+D$UGL,R1 ; Skip CUG name field ;[TD001]
2956 022520 060100 ADD R1,R0 ;[TD001]
2957 022522 010003 MOV R0,R3 ; Save address of remote task name ;[TD001]
2958 022524 116701 000024G MOVB $BFR+D$PRL,R1 ; Skip task name field ;[TD001]
2959 022530 060100 ADD R1,R0 ; Now addressing username with R0 ;[TD001]
2960 022532 116701 000025G MOVB $BFR+D$USL,R1 ; Get length of access control username ;[TD001]
2961 022536 001412 BEQ 38$ ; Branch if no username ;[TD001]
2962 022540 012702 001034' 36$: MOV #TEMP1,R2 ; Address output buffer for username ;[TD001]
2963 022544 112022 MOVB (R0)+(R2)+ ; Move user name ;[TD001]
2964 022546 SOB R1,36$ ;[TD001]
2965 022552 105012 CLRB (R2) ; Convert to ASCII string ;[TD001]

```



```

3489
3490
3491
3492 025320 012701 001034'
3493 025324 012700 000120'
3494 025330 132763 000100 000003
3495 025336 001002
3496 025340 012700 000110'
3497 025344 112021
3498 025346 001376
3499 025350 122763 000001 000004
3500 025356 001005
3501 025360
3502 025370 000404
3503
3504 025372
3505
3506 025402
3507
3508
3509
3510
3511
3512 025404
3513 025426
3514
3515 025436
3516 025453
3517
3518 025460
3519 025504
3520
3521 025512
3522 025551
3523
3524
3525 025562
3526 025615
3527
3528 025624
3529 025657
3530
3531

```

```

; DISPLAY USER INFORMATION
;
MOV #TEMP1,R1 ; GET AREA TO STORE USER STRING
MOV #LOGIN,R0 ; ASSUME LOGIN UIC
BITB #OF.RLU,0.FLG(R3) ; LOGIN UIC?
BNE 50$ ; BR IF YES
MOV #DEFAULT,R0 ; ELSE MUST BE DEFAULT
MOV (R0)+(R1)+ ; STORE USER STRING
BNE 50$
CMPB #1,0.MXC(R3) ; ONLY ONE COPY?
BNE 60$ ; BR IF NO
ERRPT$ OMSG5,MSGOUT ; DISPLAY SINGLE COPY AND USER
BR 70$ ; FINISHED

60$: ERRPT$ OMSG6,MSGOUT ; DISPLAY MULTIPLE COPIES AND USER
70$: RETURN ; FINISHED

.ENABL LC

; DISPLAY MESSAGES
;
ERMSG$ <%N% %A as of %Y%N>
ERBLK$ OMSG1,<TEMP2,TEMP1>

ERMSG$ <Object = %G%N>
ERBLK$ OMSG2,<TEMP1>

ERMSG$ < Verification = %A>
ERBLK$ OMSG3,<TEMP1>

ERMSG$ < Name = %r, Verification = %A>
ERBLK$ OMSG4,<TEMP2,TEMP1>

ERMSG$ < Copies = %A, User = %A%N>
ERBLK$ OMSG5,<SINGLE,TEMP1>

ERMSG$ < Copies = %E, User = %A%N>
ERBLK$ OMSG6,<TEMP2,TEMP1>

.DSABL LC

```

```

4038 030506 016602 000006 MOV 6(SP),R2 ; RETRIEVE STATION NUMBER
4039 030512 126001 000003 CMPB L,DLC(R0),R1 ; IS THIS A PCL LINE?
4040 030516 001056 BNE 180$ ; BR IF NO
4041 030520 032760 040000 000000 BIT #LF.RDY,L.FLG(R0) ; IS THE LINE LOADED?
4042 030526 001452 BEQ 180$ ; BR IF NO - DON'T DISPLAY IT
4043 030530 005067 000000G CLR $RBLCK ; ASSUME LINE TABLE IS NOT MAPPED
4044 030534 005767 000000G TST .MGMGE ; MAPPED SYSTEM?
4045 030540 100407 BMI 140$ ; BR IF NO
4046 030542 022760 120000 000010 CMP #120000,L.DLS(R0) ; IS LINE TABLE MAPPED?
4047 030550 101003 BHI 140$ ; BR IF NO
4048 030552 016067 000006 000000G MOV L.DLM(R0),$RBLCK ; GET LINE TABLE BIAS
4049 030560 012704 000106G 140$: GETAD L.DLS(R0),#TR.LEN ; READ IN LINE TABLE
4050 030600 014405 MOV #SBFR+R.LST+2,R4 ; POINT TO STATION TABLE LISTHEAD
4051 030604 001422 150$: MOV -(R4),R5 ; GET NEXT ENTRY
4052 030606 001422 BEQ 180$ ; BR IF END OF LIST
4053 030610 000000G 160$: GETAD R5,#S.LGTH ; READ IN TRIBUTARY STATION TABLES
4054 030626 005725 MOV #SBFR,R5 ; GET ADDRESS OF STATION TABLE
4055 030632 126502 000005 TST (R5)+ ; POINT PAST LISTHEAD
4056 030634 001402 CMPB S.LSA(R5),R2 ; IS THIS THE TRIBUTARY?
4057 030640 010504 BEQ 170$ ; BR IF YES
4058 030642 000757 MOV R5,R4 ; ELSE, CONTINUE SCAN
4059 030644 116503 000003 170$: BR 150$ ;
4060 030646 000466 210$: MOVB S.PSA(R5),R3 ; GET TRIBUTARY NUMBER
4061 030652 BR 210$ ; AND PRINT MESSAGE
4062 ;
4063 ; FIND DMP LINE TABLE
4064 ;
4065 030654 012700 015430 180$: MOV #*RDMP,R0 ; GET DMP PROCESS NAME
4066 030660 CALL FNPDV ; GET PDV INDEX
4067 030664 103473 BCS 230$ ; BR IF CAN'T FIND IT
4068 030666 016600 000012 MOV 12(SP),R0 ; RETRIEVE SLT ADDRESS
4069 030672 016600 000006 MOV 6(SP),R2 ; RETRIEVE STATION NUMBER
4070 030676 126001 000003 CMPB L,DLC(R0),R1 ; IS THIS A DMP LINE?
4071 030702 001407 BNE 240$ ; BR IF NO
4072 030704 032760 040000 000000 BIT #LF.RDY,L.FLG(R0) ; IS THE LINE LOADED?
4073 030712 001446 BEQ 210$ ; BR IF NO - DON'T DISPLAY IT
4074 030714 005067 000000G CLR $RBLCK ; ASSUME LINE TABLE IS NOT MAPPED
4075 030720 005767 000000G TST .MGMGE ; MAPPED SYSTEM?
4076 030724 100407 BMI 190$ ; BR IF NO
4077 030726 022760 120000 000010 CMP #120000,L.DLS(R0) ; IS THE LINE TABLE MAPPED?
4078 030734 101003 BHI 190$ ; BR IF NO
4079 030736 016067 000006 000000G MOV L.DLM(R0),$RBLCK ; GET LINE TABLE BIAS
4080 030744 012704 000014G 190$: GETAD L.DLS(R0),#P.LEN ; READ IN LINE TABLE
4081 030764 011405 MOV #SBFR+P.TRIB,R4 ; POINT TO STATION LINE TABLES
4082 030770 001434 200$: MOV -(R4),R5 ; GET NEXT ENTRY
4083 030772 BEQ 210$ ; BR IF END OF LIST
4084 030774 GETAD T,TRLG ; READ IN TRIBUTARY STATION TABLES
4085 031012 012704 000000G MOV #P,R4 ; GET ADDRESS OF TRIB STATION TABLES
4086 031016 126402 000005 CMPB T,LEN(R4),R2 ; IS THIS THE TRIBUTARY?
4087 031022 001352 BNE 200$ ; BR IF NO - CONTINUE SCAN
4088 031024 116703 000004G MOVB T.TRPV+$BFR,R3 ; GET TRIBUTARY NUMBER
4089 ;
4090 031030 020327 177777 210$: CMP R3,#-1 ; WAS A TRIB FOUND?
4091 031034 001407 BEQ 230$ ; BR IF NO
4092 031036 110367 150140 MOVB R3,TEMP3 ; ELSE GET TRIBUTARY NUMBER
4093 031042 000404 220$: FRRPT$ CMMSG8,MSGUUT ; DISPLAY COST,TYPE AND TRIBUTARY
4094 031052 BR 240$

```

VDIS - VNP SHOW COMMANDS
FMT CIR - FORMAT CIRCUIT-ID

MACRO V05.03b Monday 15-Jul-85 12:14^{D 7} Page 57-1

4660 034510 112720 000056
4661 034514 012702 014012
4662 034520
4663 034524 112710 000000
4664 034530
4665 034540

40\$:

MOVB #PERIOD,(R0)+
MOV #<<3*4000>!10.>,R2
CALL \$CBTA
MOVB #0,(R0)
RESRG <R3,R2,R1,R0>
RETURN

; ELSE, INSERT DELIMITER
; USE BASE 10. - FIELD WIDTH 3
; CONVERT TO ASCII
; CREATE ASCII STRING
; RESTORE REGISTERS

L\$NTBY 000052	L\$STRC 000050	NMSG10 013134R	N\$MHC1 000036	OP\$DTC= 000040
L\$NTCA 000070	L\$STIM 000051	NMSG11 013174R	N\$MHC2 000044	OP\$DUP= 000002
L\$NTFS 000102	L\$STS 000002	NMSG12 013242R	N\$PLD 000016	OP\$EST= 000100
L\$NTPK 000062	L\$ST2 000003	NMSG13 013552R	N\$PLLT 000030	OP\$EVE= 010000
L\$OMST 000002	L\$TBP 000004	NMSG14 013750R	N\$PR1 000076	OP\$FIL= 000200
L\$PCS 000112	L\$TIMC 000060	NMSG15 013770R	N\$ROA1 000022	OP\$GRP= 000100
L\$PLS 000110	L\$TIMI 000001	NMSG16 014012R	N\$ROA2 000030	OP\$HOS= 000001
L\$QCNT 000122	L\$TIMR 000000	NMSG2 012120R	N\$RTMX 000014	OP\$HTM= 000002
L\$QUEH 000124	L\$TOR 000064	NMSG3 012242R	N\$RTM1 000014	OP\$INA= 004000
L\$QUET 000130	L\$UNT 000013	NMSG4 012252R	N\$RTM2 000015	OP\$INC= 001000
L\$RTY 000007	L\$XSET 000020	NMSG5 012322R	N\$RT1 000000	OP\$KDS= 002000
L\$SLN 000004	MATCH 024762P	NMSG6 012520R	N\$RT2 000006	OP\$KDT= 000400
L\$ST 000005	MAXID = 000020	NMSG6A 012636R	N\$SLA 000016	OP\$KEV= 001000
L\$STCLZ 000044	MODTBL 015672R	NMSG6B 012734R	N\$SNOD 000012	OP\$KGR= 001000
L\$TIM 000006	MOMS1 015662R	NMSG7 013014R	N\$STCL 000112	OP\$KNT= 010000
L\$SASG= 000000	MSGOUT 035606R	NMSG8 013040R	N\$TIM 000004	OP\$KSK= 002000
L\$SDRV= 000000	MSKLEN= 000100	NMSG9 013076R	N\$TLC 000100	OP\$LCT= 000200
L\$SP11= 000001	MSPEC 000663R	NMSG12A 013314R	N\$TNC 000114	OP\$LINE= 000040
L\$SP11R= 000000	MXACCT= 000050	NMSG2A 012166R	N\$TRC 000106	OP\$LOC= 000001
L\$BABL 000062	MXCALL= 000040	NMSG5A 012404R	N\$TTCB 000110	OP\$LIST= 000002
L\$BUFU 000052	MXDTEA= 000010	NODSAV 001032R	N\$VCB 000010	OP\$LTM= 000001
L\$CHA 000034	MXHEX = 000020	NOINFD 001022R	N\$VER 000066	OP\$LVL= 000010
L\$CNTL 000024	MX25AC 000707R	NROUT 000426R	N\$XLEN 000124	OP\$MAC= 000010
L\$COST 000015	MX25PR 000672R	NSPEC 000411R	N\$SACC= 000001	OP\$MCO= 000002
L\$CRC 000030	MX25SV 000722R	NTINAM 001024R	N\$SBUF= 000001	OP\$MDE= 000400
L\$CRS 000040	MX29SV 000735R	NXTBLK 025006R	N\$SLDV= 000001	OP\$MLI= 000004
L\$CTL 000012	M\$SCRB= 000124	NXTCIR 025050R	N\$SMCP= 000001	OP\$MON= 000100
L\$CVA 177776	M\$SCRX= 000000	NXTDST 024126R	N\$SMML= 000001	OP\$MST= 000020
L\$DDM 000002	M\$SFCS= 000000	NXTDTE 021126R	N\$SMOV= 000010	OP\$NAM= 000004
L\$DDS 000004	M\$SMGE= 000000	NXTGRP 021662R	N\$SNCT= 000001	OP\$NET= 004000
L\$DISL 000042	M\$SNEI= 000000	NXTLIN 025120R	N\$SPEM= 000001	OP\$NLI= 000002
L\$DISR 000044	M\$SOVR= 000000	NXTRDT 016522R	OERR1 002510R	OP\$NOD= 000020
L\$DLC 000003	NERR1 014040R	N\$ACQ 000000	OFF 000077R	OP\$OUT= 002000
L\$DLM 000006	NF\$ACC= 100000	N\$ACTL 000032	OF\$INS= 000001	OP\$OWN= 000020
L\$DLS 000010	NF\$BLK= 000100	N\$ADJ1 000072	OF\$LOG= 000100	OP\$PAR= 000010
L\$ENA 000066	NF\$DMO= 000010	N\$ADJ2 000074	OF\$OFF= 000002	OP\$PRI= 000010
L\$FLG 000000	NF\$EVT= 040000	N\$CACH 000062	OF\$ON = 000000	OP\$RET= 010000
L\$KRBA 000016	NF\$MOU= 000040	N\$CIR 000034	OF\$SMr= 000200	OP\$RPA= 000010
L\$LEN = 000022	NF\$RST= 000002	N\$CRC 000120	OF\$PRO= 000000	OP\$SCO= 000001
L\$LNUM 000046	NF\$SCN= 000020	N\$DLA 000020	OF\$RLU= 000100	OP\$SEG= 000400
L\$MPF 000022	NF\$SHU= 000004	N\$DLY 000014	OF\$SMC= 000200	OP\$SER= 000400
L\$NAST 000055	NF\$TIM= 000200	N\$ELEN 000054	OF\$KNOWN 000037R	OP\$SIN= 000010
L\$NCIB 000012	NF\$ACC= 000001	N\$ENC 000044	OMSG1 025430R	OP\$STA= 000004
L\$NCUL= 000050	NF\$BCP= 000040	N\$ERRC 000022	OMSG2 025454R	OP\$TPA= 000020
L\$NIP 000010	NF\$END= 000200	N\$FLG 000005	OMSG3 025506R	OP\$TRI= 000040
L\$NLSE 000054	NF\$EVT= 000002	N\$FNC 000006	OMSG4 025552R	OP\$UCS= 000020
L\$NMST 000020	NF\$LLI= 000004	N\$GENQ 000052	OMSG5 025616R	OP\$USE= 000002
L\$NRSE 000053	NF\$LV2= 000100	N\$GTM 000015	OMSG6 025660R	OP\$VEC= 000040
L\$NSTA 000014	NF\$SLI= 000400	N\$HC1 000052	ON 000064R	OP\$VER= 000040
L\$OWNR 000021	NF\$SMC= 000020	N\$HC2 000056	OP\$ADD= 000200	OP\$XAC= 000001
L\$PAIC 000017	NF\$SWP= 000010	N\$HIGH 000033	OP\$ALL= 000200	OP\$XPR= 000002
L\$PAIR 000016	NKNOWN 000362R	N\$LLT 000026	OP\$CCS= 000004	OP\$XSS= 000004
L\$PLD 000007	NLMSG1 030024R	N\$LLTM 000024	OP\$CIE= 004000	OP\$XPS= 000010
L\$PLL 000012	NLMSG2 030034R	N\$LVC 000036	OP\$CON= 000400	OSPEC 000055R
L\$SCFW 000056	NLOOP 000376R	N\$LV1 000002	OP\$COP= 000001	O\$FLG 000003
L\$STA 000063	NMG13A 013716R	N\$LV2 000010	OP\$COS= 000100	O\$LEN 000012
L\$STBL 000070	NMSG1 012056R	N\$MBXQ 000050	OP\$DST= 000200	O\$LNK 000000

SYMBOL CROSS REFERENCE

CREF 04.00

SYMBOL	VALUE	REFERENCES																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
--------	-------	------------	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 22 D 10
 SYMBOL CROSS REFERENCE CREF 04.00
 SYMBOL VALUE REFERENCES

	31-2385	#33-2492	33-2493	#33-2495	33-2496	#33-2498	33-2499	#33-2501	33-2502
	#33-2504	33-2505	#33-2507	33-2508	#33-2510	33-2511	#34-2602	34-2603	#34-2607
	34-2608	#34-2610	34-2611	#34-2613	34-2614	#34-2616	34-2617	#35-2678	35-2679
	#35-2681	35-2682	#35-2684	35-2685	#37-2787	37-2788	#37-2790	37-2791	#37-2793
	37-2794	#39-2901	39-2902	#39-2906	39-2907	#40-3070	40-3071	#40-3073	40-3074
	#40-3076	40-3077	#40-3079	40-3080	#40-3082	40-3083	#40-3085	40-3086	#40-3088
	40-3089	#40-3091	40-3092	#40-3094	40-3095	#40-3097	40-3098	#40-3100	40-3101
	#40-3103	40-3104	#40-3106	40-3107	#40-3109	40-3110	#42-3189	42-3190	#42-3194
	42-3195	#48-3512	48-3513	#48-3515	48-3516	#48-3518	48-3519	#48-3521	48-3522
	#48-3525	48-3526	#48-3528	48-3529	#49-3657	49-3658	#49-3660	49-3661	#49-3663
	49-3664	#49-3666	49-3667	#50-3723	50-3724	#50-3726	50-3727	#50-3729	50-3730
	#51-3824	51-3825	#51-3827	51-3828	#51-3830	51-3831	#51-3833	51-3834	#51-3836
	51-3837	#51-3839	51-3840	#51-3842	51-3843	#51-3845	51-3846	#51-3848	51-3849
	#51-3851	51-3852	#52-3916	52-3917	#52-3919	52-3920	#53-4283	53-4284	#53-4286
	53-4287	#53-4289	53-4290	#53-4292	53-4293	#53-4295	53-4296	#53-4298	53-4299
	#53-4301	53-4302	#53-4304	53-4305	#53-4308	53-4309	#53-4311	53-4312	#54-4362
	54-4363	#54-4365	54-4366	#54-4368	54-4369	#54-4371	54-4372	#55-4529	55-4530
	#55-4532	55-4533	#55-4535	55-4536	#55-4538	55-4539	#55-4541	55-4542	#55-4544
	55-4545	#55-4547	55-4548						
\$\$\$ARG	= 000000	#6-267	6-267	#6-268	6-268	6-268			
\$\$\$GLB	= *****	6-267	6-268						
\$\$\$S	= 034164 R	#6-237	#8-353	#8-357	#8-363	#9-435	#10-548	#10-551	#11-641
		#12-764	#12-767	#12-770	#12-773	#12-777	#12-780	#12-783	#12-786
		#12-792	#12-795	#26-1831	#26-1834	#26-1837	#26-1840	#26-1843	#26-1846
		#26-1853	#26-1857	#26-1860	#26-1863	#26-1866	#26-1869	#26-1872	#26-1875
		#26-1881	#26-1887	#26-1891	#26-1894	#26-1897	#26-1900	#26-1903	#27-2137
		#29-2281	#31-2376	#31-2379	#31-2382	#31-2385	#33-2493	#33-2496	#33-2499
		#33-2575	#33-2508	#33-2511	#34-2603	#34-2608	#34-2611	#34-2614	#34-2617
		#35-2682	#35-2685	#37-2788	#37-2791	#37-2794	#39-2902	#39-2907	#40-3071
		#40-3077	#40-3080	#40-3083	#40-3086	#40-3089	#40-3092	#40-3095	#40-3098
		#40-3104	#40-3107	#40-3110	#42-3190	#42-3195	#48-3513	#48-3516	#48-3519
		#42-3526	#48-3529	#49-3658	#49-3661	#49-3664	#49-3667	#50-3724	#50-3727
		#51-3825	#51-3828	#51-3831	#51-3834	#51-3837	#51-3840	#51-3843	#51-3846
		#51-3852	#52-3917	#52-3920	#53-4284	#53-4287	#53-4290	#53-4293	#53-4296
		#53-4302	#53-4305	#53-4309	#53-4312	#54-4363	#54-4366	#54-4369	#54-4372
		#55-4535	#55-4536	#55-4539	#55-4542	#55-4545	#55-4548		
		53-4016	53-4044	53-4075					
.MGMGE	= ***** GX								
.TLGTH	= ***** GX								
.TSKHD	= ***** GX								

```
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211
```

```
                .SBTTL LOCAL SYMBOL DEFINITIONS  
                :  
                : LOCAL SYMBOL DEFINITIONS  
                :  
000006          MXCON = 6           : MAXIMUM LENGTH OF CONSOLE NAME STRING  
000006          MXMON = 6           : MAXIMUM LENGTH OF MONITOR NAME STRING  
000050          MXFIL = 40          : MAXIMUM LENGTH OF FILE NAME STRING  
077777          MXVER = 77777       : MAXIMUM FILE VERSION NUMBER  
000377          MXUNT = 377         : MAXIMUM DEVICE UNIT NUMBER VALUE  
000003          MXFILN = 3          : MAXIMUM NUMBER OF RAD50 WORDS FOR FILE NAME  
000740          CUSTEV = 480        : BEGINNING CLASS FOR CUSTOMER SPECIFIC EVENTS
```

```

738 001700 016701 000000G      MOV    $SLN,R1      ; GET SLN
739 001704 006301              ASL     R1      ; MAKE IT A WORD OFFSET
740 001706 066701 000000G      ADD    $LLCTA,R1    ; POINT AT ENTRY IN REV. MAPPING TABLE
741 001712 011101              MOV    (R1),R1    ; GET ADDRESS OF ST'ION MAPPING TABLE
742 001714 100030              BPL     105$      ; IF PL, SOMETHING IS SCREWED UP
743 001716 016702 000000G      MOV    $TRIB,R2    ; GET THE TRIBUTARY NUMBER
744 001722 006302              ASL     R2      ; MAKE IT A WORD OFFSET
745 001724 060201              ADD    R2,R1      ; AND POINT AT ENTRY
746 001726 010411 20$:        MOV    R4,(R1)      ; STORE CHANNEL NUMBER
747 001730 110061 000001      MOVB   R0,1(R1)    ; AND PDV INDEX
748 001734              RETURN
749
750      ; ERROR CONDITIONS
751
752 001736      101$:  ERRPT$  ERR36,$EROUT      ; PROCESS NOT IN SYSTEM
753 001746      102$:  ERRPT$  ERR35,$EROUT      ; LINE IN WRONG STATE
754 001756      103$:  ERRPT$  ERR21,$EROUT      ; LINE NOT IN SYSTEM
755 001766      104$:  ERRPT$  ERR37,$EROUT      ; NO FREE CHANNELS
756 001776      105$:  ERRPT$  ERR22,$EROUT      ; SLT AND REVERSE TABLE MISMATCH

```



```

1251          .SBTTL  CHKEXE - CHECK IF EXECUTOR NODE
1252      ;+
1253      *** - CHKEXE - CHECK EXECUTOR NODE
1254
1255      INPUTS:
1256          $VFLAG = VNP PROCESSING FLAGS
1257
1258      OUTPUTS:
1259          Z-BIT = SUCCESS/FAILURE
1260
1261      :-
1262
1263      CHKEXE:
1264      003730      BIT      #VF.ADD,$VFLAG          ; WAS NODE ADDRESS OR NAME GIVEN ?
1265      003730      032767      000000G 000000G      BEQ      10$          ; IF EQ, NODE NAME WAS GIVEN
1266      003736      001414
1267      003740      GETRV    $DECT,#D$END          ; READ DECNET HOME BLOCK
1268
1269      003760      026767      000012G 000014G      CMP      RQB+LR.ADD,$BFR+D$LNUM ; IS THIS THE EXECUTOR NODE ADDRESS ?
1270      003766      000413      20$          BR      20$          ; RETURN Z-BIT
1271      003770      026767      000012G 000006G 10$:    CMP      RQB+LR.NAM,$BFR+D$LNAM ; IS THIS THE EXECUTOR NODE NAME ?
1272      003776      001007      20$          BNE      20$          ; IF NE, NO
1273      004000      026767      000014G 000010G      CMP      RQB+LR.NAM+2,$BFR+D$LNAM+2 ; IS IT REALLY THE EXECUTOR ?
1274      004006      001003      20$          BNE      20$          ; IF NE, NO
1275      004010      026767      000016G 000012G      CMP      RQB+LR.NAM+4,$BFR+D$LNAM+4 ; LAST CHANCE
1276      004016      20$:    RETURN
1277
1278
1279      ;+
1280      NSP SEGMENT SIZE MODIFICATION
1281
1282      INPUTS:
1283          RQB+LO.SEG = SEGMENT SIZE
1284
1285      OUTPUTS:
1286          DECNET HOME BLOCK UPDATED
1287
1288      :-
1289
1290      $VSEG:: GETRV    $DECT,#D$END          ; READ HOME BLOCK
1291      004020      016767      000054G 000036G      MOV      LO.SEG+RQB,$BFR+D$SEG ; STORE SEGMENT SIZE
1292      004046      PUTRC          ; WRITE HOME BLOCK
1293
1294      004052      RETURN
1295
1296
1297      ;+
1298      INCOMING TIMER
1299
1300      INPUTS:
1301          RQB+LO.INC = INCOMING TIMER
1302
1303      OUTPUTS:
1304          DECNET HOME BLOCK UPDATED
1305
1306      :-
1307      $VINC:: GETRV    $DECT,#D$END          ; READ HOME BLOCK
  
```

```

1780      .SBTTL  GTSLT - FIND SLT ADDRESS AND STATION NUMBER
1781      :+
1782      : **-$GTSLT-FIND SLT ADDRESS AND STATION NUMBER
1783      :
1784      :   FIND THE SYSTEM LINE TABLE ADDRESS AND STATION NUMBER ASSOCIATED
1785      :   WITH A GIVEN PDV AND CHANNEL #.
1786      :
1787      : INPUTS:
1788      :   R2 - CHANNEL NUMBER
1789      :
1790      : OUTPUTS:
1791      :   R2 - POINTER TO SLT
1792      :   R3 - STATION NUMBER
1793      : -
1794
1795      006044 010046      GTSLT:  MOV    R0,-(SP;      ; SAVE R0
1796      006046 010146      MOV    R1,-(SP;      ; SAVE R1
1797      006050 042702 177400      BIC    #^C<377>,R2      ; CLEAR OUT EXTRANEIOUS BITS IN CHANNEL NUMBER
1798      006054 010203      MOV    R2,R3      ; SAVE CHANNEL NUMBER
1799      006056      CALL    CHKXPT      ; FIND XPT'S PDV ADDRESS
1800      006062 103417      BCS    10$      ; IF CS, NO XPT
1801      006064 006303      ASL    R3      ; FORM WORD INDEX FROM CHANNEL NUMBER
1802      006066 060003      ADD    R0,R3      ; INDEX INTO PDV
1803      006070 016302 000000G      MOV    Z,MAP(R3),R2      ; GET THE SLN/STATION #
1804      006074 010203      MOV    R2,R3      ; SAVE FOR LATER
1805      006076 042702 177400      BIC    #^C<377>,R2      ; ISOLATE SYSTEM LINE #
1806      006102 006302      ASL    R2      ; FORM WORD OFFSET
1807      006104 066702 000000G      ADD    $SLTMA,R2      ; POINT INTO SYSTEM LINE INDEX TABLE
1808      006110 011202      MOV    (R2),R2      ; GET SLT ADDRESS
1809      006112 000303      SWAB   R3      ; GET STATION ADDRESS TO LOW BYTE
1810      006114 042703 177400      BIC    #^C<377>,R3      ; ISOLATE STATION #
1811      006120 000241      CLC      ; INDICATE SUCCESS
1812      006122 012601      10$:  MOV    (SP)+,R1      ; RESTORE R1
1813      006124 012600      MOV    (SP)+,R0      ; RESOTRE R0
1814      006126      RETURN

```

```

2401 011512          CALL      MOVE          ; STORE DESTINATION FIELD
2402 011516          MOV       #RQB+LR,UIC,R2 ; GET ADDRESS OF INPUT BUFFER
2403 011522          MOV      (R2)+,R3        ; GET LENGTH OF UIC FIELD
2404 011524          MOV      R3,(R1)+        ; STORE LENGTH OF UIC FIELD
2405 011526          CALL      MOVE          ; STORE UIC FIELD
2406 011532          MOV      #RQB+LR,PAS,R2  ; GET ADDRESS OF INPUT BUFFER
2407 011536          MOV      (R2)+,R3        ; GET LENGTH OF PASSWORD FIELD
2408 011540          MOV      R3,(R1)+        ; STORE LENGTH OF PASSWORD FIELD
2409 011542          CALL      MOVE          ; STORE PASSWORD FIELD
2410 011546          MOV      #RQB+LR,ACO,R2  ; GET ADDRESS OF INPUT BUFFER
2411 011552          MOV      (R2)+,R3        ; GET LENGTH OF ACCOUNT FIELD
2412 011554          MOV      R3,(R1)+        ; STORE LENGTH OF ACCOUNT FIELD
2413 011556          CALL      MOVE          ; STORE ACCOUNT FIELD
2414 011562          MOV      (SP)+,R1        ; RESTORE SIZE OF ALIAS BLOCK
2415
2416          ; INSERT ENTRY INTO ALIAS LIST
2417
2418 011564          MOV      $HMBUF+D$ANN,$BFR ; INSERT ENTRY INTO FRONT OF LIST
2419 011572          SWTCHO          ; USE DEFAULT OUTPUT BUFFER ($BFR)
2420
2421 011600          MOV      R0,-(SP)          ; ...
2422 011602          CEACC$          ; CONVERT TO MAPPED ADDRESS
2423 011612          PUTAD          R0,R1        ; WRITE OUT ALIAS BLOCK
2424 011626          MOV      (SP)+,$HMBUF+D$ANN ; SET UP NEW FIRST ALIAS POINTER
2425
2426 011632          SWTCHO          #HMBUF      ; USE TEMPORARY BUFFER
2427 011640          PUTRC          $DECPT,#D$END ; WRITE BACK DECNET HOME BLOCK
2428 011660          SWTCHO          ; RESET DEFAULT BUFFER
2429
2430 011666          50$:          RETURN
2431
2432
2433          ; ERROR CONDITIONS
2434
2435
2436 011670          101$:          ERRPT$ ERR24,$EROUT ; ALIAS NOT IN SYSTEM
2437 011700          102$:          ERRPT$ ERR25,$EROUT ; ALIAS BLOCK ALLOCATION FAILURE
2438 011710          103$:          ERRPT$ ERR19,$EROUT ; NODE NOT IN SYSTEM
2439
2440          .DSABL          LSB
2441
2442
2443          ; MOVE SUBROUTINE
2444
2445 011720          MOVE:          TST          R3          ; ANY DATA ?
2446 011722          BEQ          20$          ; IF EQ, NO
2447 011724          MOV      (R2)+,(R1)+      ; ELSE, STORE BYTE
2448 011726          DEC          R3          ; ANY MORE ?
2449 011730          BGT          10$          ; IF GT, YES
2450 011732          20$:          RETURN

```

```

2922                                     .SBTTL  GETLEN - GET LENGTH OF SINK NAME STRING
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938 013572 012700 000035G
2939 013576 005046
2940 013600 105710
2941 013602 001406
2942 013604 005200
2943 013606 005216
2944 013610 020116
2945 013612 103372
2946 013614 005726
2947 013616 000404
2948 013620 012604
2949 013622 001402
2950 013624 000241
2951 013626 000401
2952 013630 000261
2953 013632

;+
; GETLEN - GET LENGTH OF SINK NAME STRING
; INPUTS:
;   RQB+LO.SNM - SINK NAME
;   R1         - MAXIMUM NUMBER OF CHARACTERS FOR SINK NAME
; OUTPUTS:
;   CARRY      - SUCCESS/FAILURE
;   R0         - POINTS TO BYTE AFTER END OF SINK NAME
;   R4         - NUMBER OF BYTES IN SINK NAME
;-
GETLEN: MOV    #RQB+LO.SNM,R0    ; POINT TO SINK NAME
        CLR    -(SP)            ; USE STACK AS COUNTER FOR STRING LENGTH
10$:    TSTB   (R0)              ; END OF STRING?
        BEQ    20$              ; BR IF NO
        INC    R0               ; POINT TO NEXT CHARACTER
        INC    (SP)             ; UPDATE STRING LENGTH
        CMP    R1,(SP)          ; VALID LENGTH FOR MONITOR NAME?
        BHS    10$              ; BR IF YES
        TST    (SP)+            ; ELSE CLEAN UP STACK
        BR     30$              ; ELSE INDICATE ERROR
20$:    MOV    (SP)+,R4          ; GET NUMBER OF CHARACTERS IN NAME
        BEQ    30$              ; BR IF ZERO
        CLC                     ; INDICATE SUCCESS
        BR     40$              ; AND EXIT
30$:    SEC                     ; INDICATE ERROR
40$:    RETURN

```

1260	006700	005001		CLR	R1	:	GET THE TRIBUTARY NUMBER
1261	006702	156301	000021	BISB	F, LIN+1(R3), R1	:	...
1262	006706	005002		CLR	R2	:	SUPPRESS LEADING ZEROES
1263	006710			CALL	\$CBDMG	:	CONVERT IT TO ASCII
1264							
1265	006714	112720	000042	30\$:	MOVB	#''', (R0)+	: STORE TRAILING DELIMITER
1266	006720	112720	000040		MOVB	#SPACE, (R0)+	: STORE SPACES
1267	006724	112720	000040		MOVB	#SPACE, (R0)+	: ...
1268	006730	005726			TST	(SP)+	: CLEAN UP STACK
1269	006732	005726			TST	(SP)+	: ...
1270	006734				RETURN		

```

1872 013132          ERBLK$ NMSG10
1873
1874 013136          ERMSG$ < Verification state = %A%N>
1875 013172          ERBLK$ NMSG11,<TEMP2>
1876
1877 013200          ERMSG$ < Host = %D, Maximum links = %D>
1878 013240          ERBLK$ NMSG12,<TEMP3,TEMP1>
1879
1880 013250          ERMSG$ < Host = %D,%D, Maximum links = %D>
1881 013313          ERBLK$ NMSG12A,<TEMP7,TEMP3,TEMP1>
1882
1883
1884 013324          ERMSG$ < Segment buffer size = %D,Retransmit factor = %D%N>
1885 013410          ERMSG$ < Incoming timer = %D, Outgoing timer = %D%N>
1886 013472          ERMSG$ < Inactivity timer = %D, Delay factor = %D>
1887 013550          ERBLK$ NMSG13,<TEMP2,TEMP3,TEMP4,TEMP5,TEMP6,TEMP7>
1888
1889 013570          ERMSG$ < Delay weight = %D, Link service threshold = %D%N>
1890 013660          ERMSG$ : Input packet limiter = %D>
1891 013714          ERBLK$ NMSG13A,<TEMP2,TEMP3,TEMP4>
1892
1893 013726          ERMSG$ <%NExecutor node%N>
1894 013747          ERBLK$ NMSG14
1895
1896 013752          ERMSG$ < State = %A>
1897 013767          ERBLK$ NMSG15,<TEMP1>
1898
1899 013774          ERMSG$ < Type = DLX>
1900 014011          ERBLK$ NMSG16
1901
1902 014014          ERMSG$ <Node not in system>
1903 014036          ERBLK$ NERR1
1904
1905          .DSABL LC

```

```

2413          .SBTTL DMXACN - Display X25-ACCESS networks          [TD001]
2414          +                                                    [TD001]
2415          *** DMXACN - Display X25-ACCESS networks              [TD001]
2416          :                                                    [TD001]
2417          : INPUTS                                              [TD001]
2418          : $PSIPT - Address of PSI homeblock                    [TD001]
2419          : $OPTON - Command option flags                        [TD001]
2420          : RQB - Request block                                  [TD001]
2421          :                                                    [TD001]
2422          : OUTPUTS                                             [TD001]
2423          : Information displayed on user's terminal            [TD001]
2424          :                                                    [TD001]
2425          DMXACN:                                              [TD001]
2426          BIT    #PF.XGA,$PFLAG          : Check for host access system [TD001]
2427          BEQ    92$                     : Branch if not host         [TD001]
2428          MOV    $PSIPT,R0                : Get address of listhead    [TD001]
2429          ADD    #H$RNW,R0               [TD001]
2430          GETRV  R0,#2                   : Read in listhead           [TD001]
2431          :                                                    [TD001]
2432          10$: CLR    R0                  : Assume no network specified [TD001]
2433          BIT    #OP$KNT,$OPTON          : Are we looking for KNOWN NETWORKS? [TD001]
2434          BNE    20$                     : Branch if yes              [TD001]
2435          MOV    #RQB+LO.NET,R0          : Look for specified network    [TD001]
2436          20$: MOV    #R$NWL,R1          : Length of remote network block [TD001]
2437          MOV    #R$NWK,R2              : Offset of network name       [TD001]
2438          MOV    #R$NML,R3              : Offset of network name length [TD001]
2439          CALL   FNDBLK                  : Look for matching block     [TD001]
2440          BCS    90$                     : Branch if not found         [TD001]
2441          BIS    #F.FND,FLAGS            : Indicate block found        [TD001]
2442          :                                                    [TD001]
2443          MOV    #TEMP1,R2                : Copy network name to message [TD001]
2444          TST    R1                      [TD001]
2445          BEQ    35$                     [TD001]
2446          30$: MOVB  (R0)+(R2)+          :                               [TD001]
2447          SOB    R1,30$                  [TD001]
2448          35$: CLRB  (R2)                [TD001]
2449          ERRPT$ ACNS1,MSGOUT            : Show network name           [TD001]
2450          MOV    #6,R1                   : Copy node name to message    [TD001]
2451          MOV    #TEMP1,R2                [TD001]
2452          MOV    #B$FR+R$NOD,R0          [TD001]
2453          45$: MOVB  (R0)+(R2)+          :                               [TD001]
2454          SOB    R1,45$                  [TD001]
2455          CLRB  (R2)                    [TD001]
2456          ERRPT$ ACNS2,MSGOUT            : Show node name              [TD001]
2457          MOVB  #B$FR+R$USL,R1           : Get length of user id       [TD001]
2458          BEQ    60$                     : Branch if no user name      [TD001]
2459          MOV    #B$FR+R$USR,R0          : Copy user id to message     [TD001]
2460          MOV    #TEMP1,R2                [TD001]
2461          55$: MOVB  (R0)+(R2)+          :                               [TD001]
2462          SOB    R1,55$                  [TD001]
2463          CLRB  (R2)                    [TD001]
2464          ERRPT$ ACNS3,MSGOUT            : Show user id                [TD001]
2465          MOVB  #B$FR+R$PAL,R1           : Get length of password      [TD001]
2466          BEQ    70$                     : Branch if no password      [TD001]
2467          ERRPT$ ACNS4,MSGOUT            : Display 'password set' message [TD001]
2468          MOVB  #B$FR+R$ACL,R1           : Get length of account       [TD001]
2469          BEQ    80$                     : Branch if no account        [TD001]

```

```

2966 022554      116701 000026G      38$:  ERRPT$  DSMS10,MSGOUT      ; Display username      [TD001]
2967 022564      116701 000026G      MOV$  $BFR+$SPAL,R1      ; Get length of access password [TD001]
2968 022570      001405      BEQ  40$      ; Branch if not set      [TD001]
2969 022572      060100      ERRPT$  DSMS11,MSGOUT      ; Indicate password set [TD001]
2970 022602      116701 000027G      ADD  R1,R0      ; Step to account field [TD001]
2971 022604      001412      MOV$  $BFR+$SACL,R1      ; Get length of account field [TD001]
2972 022610      012702      BEQ  44$      ; Branch if no account [TD001]
2973 022612      112022      MOV  #TEMP1,R2      ; Move account to output buffer [TD001]
2974 022616      105012      42$:  MOV$  (R0)+(R2)+      ; [TD001]
2975 022620      010004      SOB  R1,42$      ; [TD001]
2976 022624      005067      CLRB  (R2)      ; [TD001]
2977 022626      156170      ERRPT$  DSMS12,MSGOUT      ; Display account [TD001]
2978 022636      005067      MOV  R0,R4      ; Save address of call data field [TD001]
2979 022640      156767      CLR  TEMP1      ; Get priority [TD001]
2980 022644      000002G 156162      BISB  $BFR+$SPRI,TEMP1      ; [TD001]
2981 022652      116767      MOV$  $BFR+$SOBJ,TEMP2      ; Get object number [TD001]
2982 022660      001016      BNE  48$      ; Branch if object is numbered [TD001]
2983 022662      012702      MOV  #TEMP2,R2      ; Move object name to output buffer [TD001]
2984 022666      116701 000024G      MOV$  $BFR+$SPRL,R1      ; ... unless length is zero [TD001]
2985 022672      001403      BEQ  47$      ; [TD001]
2986 022674      112322      46$:  MOV$  (R3)+(R2)+      ; [TD001]
2987 022676      105012      SOB  R1,46$      ; [TD001]
2988 022702      000437      47$:  CLRB  (R2)      ; [TD001]
2989 022704      000437      ERRPT$  DSMS2C,MSGOUT      ; Write object name and priority [TD001]
2990 022714      000437      BR  55$      ; [TD001]
2991 022716      000437      48$:  ERRPT$  DSMS2B,MSGOUT      ; Write object number and priority [TD001]
2992 022726      000437      BR  55$      ; [TD001]
2993      ; [TD001]
2994      ; Display object and priority for old-format block [TD001]
2995      ; [TD001]
2996 022730      012704 000032G      50$:  MOV  #BFR+$VBL,R4      ; Set address of old-format call data [TD001]
2997 022734      005067      CLR  TEMP1      ; Get priority [TD001]
2998 022740      156767      BISB  $BFR+$SPRI,TEMP1      ; [TD001]
2999 022746      116767      MOV$  $BFR+$SOBJ,TEMP2      ; Get object number [TD001]
3000 022754      001405      BEQ  52$      ; Branch if object named [TD001]
3001 022756      000412      ERRPT$  DSMS2B,MSGOUT      ; Display priority and object number [TD001]
3002 022766      016767      BR  55$      ; [TD001]
3003 022770      000004G 156064      52$:  MOV  $BFR+$SNAM,TEMP2      ; Get RAD50 task name [TD001]
3004 022776      016767      MOV  $BFR+$SNAM+2,TEMP2+2      ; [TD001]
3005 023004      000006G 156060      ERRPT$  DSMS2A,MSGOUT      ; Display priority and task name [TD001]
3006      ; [TD001]
3007      ; Display call mask/value and DTE [TD001]
3008      ; [TD001]
3009 023014      010401      55$:  MOV  R4,R1      ; Get address of start of call data [TD001]
3010 023016      116700      MOV$  D$D$ATL+$BFR,R0      ; Get length of call mask/call value [TD001]
3011 023022      001416      BEQ  58$      ; Branch if none [TD001]
3012 023024      012703 001034'      SAVRG <R0>      ; Save byte count [TD001]
3013 023026      012703 001034'      MOV  #TEMP1,R3      ; Point to temporary buffer [TD001]
3014 023032      012703 001062'      CALL HEXASC      ; Convert to asic string [TD001]
3015 023036      012703 001062'      RESRG <R0>      ; Restore byte count [TD001]
3016 023040      012703 001062'      MOV  #TEMP2,R3      ; Point to temporary buffer [TD001]
3017 023044      012703 001062'      CALL HEXASC      ; Convert to ASCII string [TD001]
3018 023050      012703 001062'      ERRPT$  DSMS3,MSGOUT      ; Display call mask/call value [TD001]
3019      ; [TD001]
3020 023060      116702 000031G      58$:  MOV$  D$D$TEL+$BFR,R2      ; Get length of DTE address [TD001]
3021 023064      001410      BEQ  60$      ; Branch if no DTE address [TD001]
3022 023066      012703 001034'      MOV  #TEMP1,R3      ; Point to temporary buffer [TD001]

```


.SBTTL DISALI - DISPLAY ALIAS INFO

DISALI - DISPLAY ALIAS INFORMATION

INPUTS:

R3 = ADDRESS OF ALIAS NAME BLOCK
R4 = 0 IF HEADER IS TO BE DISPLAYED ALONG WITH ALIAS NAME INFORMATION,
OTHERWISE DISPLAY ONLY ALIAS NAME INFORMATION
IF HEADER MESSAGE IS TO BE DISPLAYED:
TEMP1 = SECOND ASCIZ STRING OF HEADER MESSAGE
TEMP2 = FIRST ASCIZ STRING OF HEADER MESSAGE

OUTPUTS:

THE ALIAS NAME INFORMATION IS DISPLAYED ON THE USER'S TERMINAL

R0, R1, R2 DESTROYED
ALL OTHER REGISTERS PRESERVED

```

DISALI: TST      R4                ; PRINT HEADER?
        BNE     10$              ; BR IF NO
        ERRPT$  AMSG1,MSGOUT      ; DISPLAY HEADER

; DISPLAY ALIAS NAME
10$:    MOV      #TEMP1,R1         ; STORAGE FOR NAME
        MOV      R3,R0            ; GET ADDRESS OF ALIAS NAME BLOCK
        ADD      #A.NAM,R0        ; POINT TO ALIAS NAME
        .REPT    3
        MOV      (R0)+,(R1)+      ; STORE THE NAME
        .ENDR
30$:    MOV      #0,(R1)          ; CREATE ASCIZ STRING
        ERRPT$  AMSG2,MSGOUT      ; DISPLAY ALIAS NAME

; GET SCOPE INFORMATION
        MOV      #-1,TEMP1        ; ASSUME SCOPE IS GLOBAL
        TST      A.UCB(R3)        ; IS SCOPE GLOBAL?
        BEQ      40$              ; BR IF YES
        SWTCHI  #TEMP2            ; USE TEMP2 AS TEMP INPUT BUFFER
        MOV      #TEMP2,R2        ; GET ADDRESS OF TEMP INPUT BUFFER
        GETRV   A.UCB(R3),#U.LEN  ; READ IN UCB
        GETRV   U.DCB(R2),#D.LEN  ; READ IN DCB
        MOV      D.NAM(R2),TEMP3   ; SAVE DEVICE NAME
        MOV      A.UCB(R3),R0      ; GET UCB ADDRESS
        SUB     D.UCB(R2),R0       ; CALCULATE OFFSET
        MOV     D.UCBL(R2),R1      ; GET LENGTH OF UCB
        CALL    $DIV              ;
        ADD     D.UNIT(R2),R0      ; POINT TO CORRECT UNIT
        BIC     #177400,R0        ;
        MOV     R0,TEMP1          ; STORE UNIT NUMBER
        MOV     #0,3+TEMP3        ; CREATE ASCIZ STRING
        SWTCHI  $BUFR IS INPUT BUFFER

```

```

4095 031054          230$:  ERRPT$  CMSG9,MSGOUT          ; DISPLAY COST AND TYPE
4096
4097
4098
4099          ; GET CIRCUIT OWNER
4100
4101 031064 016600 000012      240$:  MOV      12(SP),R0          ; RESTORE R0
4102 031070 016601 000010      MOV      10(SP),R1          ; RESTORE R1
4103 031074 016602 000006      MOV      6(SP),R2           ; RESTORE R2
4104 031100 116003 000021      MOV      L,NMST+1(R0),R3      ; ASSUME OWNER IS IN LINE SLT
4105 031104 105760 000014      TSTB     L,NSTA(R0)          ; MULTIPOINT CIRCUIT?
4106 031110 001412      BEQ      260$                      ; BR IF NO
4107 031112 011604      MOV      (SP),R4                   ; GET MULTIPOINT FLAG TABLE ADDRESS
4108 031114 032760 000400 000000  BIT     #LF,BRO,L.FLG(R0)  ; ETHERNET CIRCUIT ?
4109 031122 001403      BEQ      250$                      ; IF NOT - BRANCH
4110 031124 010004      MOV      R0,R4                     ; GET MULTIPOINT FLAG TABLE ADDRESS
4111 031126 062704 000022      ADD      #L,MFP,R4           ; ...
4112 031132 116403 000003      250$:  MOV      S,NMST+1(R4),R3  ; GET OWNER FROM TRIB EXTENSIONS
4113 031136 005703      260$:  TST      R3                 ; IS THERE AN OWNER SET?
4114 031140 001431      BEQ      290$                      ; BR IF NO
4115 031142 012700 001202'      MOV      #TEMP3,R0          ; STORAGE FOR CIRCUIT OWNER
4116 031146 066703 000000G      ADD      $PDVTA,R3          ; INDEX INTO PDV TABLE
4117 031152 011303      MOV      (R3),R3                   ; GET PDV ADDRESS
4118 031154 016301 000000G      MOV      Z,NAM(R3),R1        ; GET OWNER NAME IN RAD50
4119 031160      CALL     $CSTA                               ; CONVERT IT TO ASCII
4120 031164 124027 000040      CMPB     -(R0),#SPACE       ; DELETE TRAILING BLANKS
4121 031170 001004      BNE      270$                      ; ...
4122 031172 124027 000040      CMPB     -(R0),#SPACE       ; ...
4123 031176 001001      BNE      270$                      ; ...
4124 031200 005300      DEC      R0                        ; ...
4125 031202 112760 000000 000001 270$:  MOV      #0.1(R0)    ; CREATE ASCII STRING
4126 031210 016600 000012      MOV      12(SP),R0          ; GET SLT ADDRESS
4127 031214          280$:  ERRPT$  CMSG7,MSGOUT          ; DISPLAY CIRCUIT OWNER
4128
4129          ; GET CIRCUIT STATE
4130
4131 031224 016600 000012      290$:  MOV      12(SP),R0          ; RESTORE R0
4132 031230 016601 000010      MOV      10(SP),R1          ; RESTORE R1
4133 031234 012705 001034'      MOV      #TEMP5,R5          ; STORAGE FOR CIRCUIT STATE
4134 031240 012704 000246'      MOV      #PCLEAR,R4         ; ASSUME IT IS CLEARED
4135 031244 032760 040000 000000 300$:  BIT     #LF,RDY,L.FLG(R0)  ; IS LINE CLEARED?
4136 031252 001431      BEQ      350$                      ; BR IF YES
4137 031254 012704 000077'      MOV      #OFF,R4           ; ASSUME OFF
4138 031260 105760 000014      TSTB     L,NSTA(R0)          ; MULTIPOINT CIRCUIT?
4139 031264 001005      BNE      310$                      ; BR IF YES
4140 031266 032760 002000 000000  BIT     #LF,ENA,L.FLG(R0)  ; IS STATE OFF?
4141 031274 001420      BEQ      350$                      ; BR IF YES
4142 031276 000415      BR      340$                      ; ELSE STATE IS ON
4143 031300 032760 000400 000000 310$:  BIT     #LF,BRO,L.FLG(R0)  ; BROADCAST LINE ?
4144 031306 001404      BEQ      320$                      ; IF NO - BRANCH
4145 031310 010003      MOV      R0,R3                     ; COPY SLT ADDRESS
4146 031312 062703      ADD      #L,MFP,R3                 ; CALCULATE STATION TABLE ADDRESS
4147 031316 000401      BR      330$                      ; CONTINUE
4148
4149 031320 011603      320$:  MOV      (SP),R3             ; GET MULTIPOINT STATION ADDRESS
4150 031322 132763 000100 000000 330$:  BITB     #SF,ENA,S.FLG(R3)  ; IS STATE OFF?
4151 031330 001402      BEQ      350$                      ; BR IF YES

```

```

4667                                     .SBTTL  GTEXEC - GET EXECUTOR ADDRESS AND NAME
4668
4669                                     ;+
4670                                     ; GTEXEC - GET EXECUTOR ADDRESS AND NAME
4671                                     ;
4672                                     ; INPUTS: NONE
4673                                     ;
4674                                     ; OUTPUTS:
4675                                     ; TEMP1 = EXECUTOR NODE ADDRESS (BINARY)
4676                                     ; TEMP2 = EXECUTOR NODE NAME (ASCIZ)
4677                                     ;
4678                                     ; RO,R1 DESTROYED
4679                                     ; -
4680
4681
4682 034542                                     GTEXEC: GETRV  $DECPT,$D$END          ; READ DECNET HOME BLOCK
4683 034562 016767 000014G 144244             MOV    $BFR+$D$LNAM,TEMP1      ; GET EXECUTOR ADDRESS
4684 034570 012700 000000G                   MOV    #$BFR,R0          ; POINT TO EXECUTOR NODE NAME
4685 034574 062700 000006                   ADD     #$D$LNAM,R0        ; ...
4686
4687 034600 012701 001062'                   MOV    #TEMP2,R1          ; POINT TO STORAGE FOR EXECUTOR NAME
4688 000003                                     .REPT    3
4689                                     MOV    (R0)+,(R1)+          ; STORE THE NAME
4690
4691 034612 124127 000040 10$:                 .ENDR
4692 034616 001775                                     CMPB   -(R1),#SPACE      ; DELETE BLANK FILLS
4693 034620 112761 000000 000001             BEQ    10$              ; ...
4694 034626                                     MOVB   #0,1(R1)         ; CREATE ASCIZ STRING
4695                                     RETURN
4696

```

VDIS - VNP SHOW COMMANDS

VDIS	VNI
Symbol	table

O.MXC	000004	PVMSG4	033070R
O.NAM	000006	PX\$BLK=	000040
O.TYP	000002	PX\$DLM=	000200
ACTIV	000175R	PX\$SVC=	000100
PCLEAR	000246R	PX\$CHN	000004
PERIOD=	000056	P\$CNT	000005
PERRT	003444R	P\$CTR	000034
PF\$BLK=	020000	P\$DRCT	000015
PF\$CLC=	010000	P\$DRTR	000036
PF\$DLM=	100000	P\$FLG	000012
PF\$EIP=	000002	P\$FWD	000030
PF\$ENB=	000001	P\$ICCB	000046
PF\$FA1=	004000	P\$IPL	000014
PF\$FM1=	000100	P\$LCD	000002
PF\$FM2=	000200	P\$LEN	000052
PF\$OFF=	000000	P\$LST	000000
PF\$RM1=	000020	P\$NRNI	000040
PF\$RM2=	000040	P\$OCCB	000050
PF\$RVR=	000010	P\$PQ	000006
PF\$STA	000004	P\$PSXZ	000044
PF\$SVC=	040000	P\$RMX1	000016
PF\$UP=	000004	P\$RMX2	000020
PF.XDF=	*****	P\$RPMI	000042
PF.XGA=	*****	P\$RTIM	000003
	GX	P\$STA1	000022
	GX	P\$STA2	000023
PKNOWN	000216R	P\$TIM	000010
PLIPDV	001030R	P\$TSC	000026
PMSG1	026650R	P\$TSIZ	000024
PMSG2	026676R	P\$TYP	000001
PMSG3	026722R	P\$UP45=	000000
PREVNT=	001205R	P\$WURD=	000000
PR\$BED=	000200	P.ACTC	000012
PR\$BEU=	000100	P.CFNC	000034
PR\$BRD=	000040	P.CHAR	000042
PR\$BRU=	000020	P.CONT	000067
PR\$DOWN=	000002	P.DEDT	000100
PR\$LCC=	000010	P.DELT	000076
PR\$MOP=	000004	P.DISL	000006
PR\$UP=	000001	P.DISR	000004
PSPEC	0000236R	P.DVRF	000047
PS\$CHR=	000016	P.DVTM	000064
PS\$FA1=	000014	P.ECCB	000064
PS\$NTI=	000006	P.ENBC	000011
PS\$OFF=	000000	P.HDER	000056
PS\$STR=	000002	P.INPQ	000036
PS\$UP=	000012	P.LEN =	000154
PS\$VER=	000010	P.LERR	000054
PS\$WT=	000004	P.LINE	000010
PT\$BRD=	000200	P.MODE	000060
PT\$DRT=	000100	P.MODI	000061
PT\$END=	000004	P.MTER	000057
PT\$LV1=	000002	P.OFFS=	000046
PT\$LV2=	000001	P.PARM	000070
PT\$PH3=	000010	P.PIND	000066
PT\$XAR=	000020	P.POLT	000102
PVMSG1	032756R	P.RAOC	000020
PVMSG2	033006R		
PVMSG3	033044R		

P.RASO	000022
P.RBFC	000021
P.RERR	000052
P.S.EL	177776
P.SPFC	000104
P.STAT	000002
P.STMT	000074
P.TBAD	000062
P.TIMC	000050
P.TIME	000001
P.TIMR	000000
P.TRIB	000014
P.VECT	177732
P.VEC1	177752
P.XAQC	000026
P.XAWQ	000030
P.XBFC	000027
QF\$ACT	000001
QF\$ALL	000002
QF\$KNO	000004
QF\$LOO	000010
Q\$ACTC	000004
Q\$AUC	000006
Q\$CLEN	000006
Q\$CTIM	000000
Q\$ICRE	000014
Q\$LEN	000016
Q\$MXAC	000012
Q\$MXVC	000002
Q\$TCLZ	000010
Q\$SOPT	000010
Q.IOAE	000012
Q.IOEF	000006
Q.IOFN	000002
Q.IOLU	000004
Q.IOPL	000014
Q.IOPR	000007
Q.IOSB	000010
RCVLNG	000044
RF.CTL	000003
RF.LD1	000040
RF.LD2	000100
RF.LOO	100000
RF.TIM	177400
RF.TMO	000400
RF.WFC	000200
RF.WTD	000020
RF.WTM	000030
RF.WTS	000010
ROUTNG	000416R
RGB	***** GX
RX.NXM	000020
RX.OVR	000010
R\$ACC	000064
R\$ACL	000063
R\$DAL	000013
R\$DIE	000014

```

R$LEN      000024
R$LNK      000000
R$NAM      000002
R$NMAL     000002
R$NMD      000023
R$NWK      000003
R$NWL      000104
R$NXT      000000
R$OWN      000010
R$PAL      000052
R$PAS      000053
R$USL      000031
R$USR      000032
R$SDER=    000000
R$SK11=    000001
R$SSND=    000000
R$S11M=    000000
R.ADD      000010
R.CSR      000056
R.F.LAG     000012
R.FLG      000064
R.LEN      000014
R.LNK      000000
R.LST      000104
R.MAP      000110
R.NAM      000002
R.QUE      000062
R.SLN      000070
R.SRV      000060
R.STA      000066
R.STBL      000210
SE.IRS=    004000
SE.NRS=    002000
SE.RBS=    001000
SE.RBU=    004040
SE.RCH=    000001
SE.RDC=    000002
SE.RRR=    000004
SE.SBS=    002000
SE.SBU=    000100
SE.SDC=    000020
SE.SHC=    000010
SE.SRR=    000040
SF.ACT=    000200
SF.ENA=    000100
SF.LPB=    000004
SF.MFL=    000040
SF.PAC=    000020
SF.REA=    000010
SF.SER=    000001
SF.SVC=    000002
SF.UNL=    000040
SHUT       000103R
SINGLE      000126R
SLASH=     000057
SMSG1      002100R
SMSG2      002242R

```

```

SPACE = 000040
SUMMARY 000007R
SYNNERR 002270R
SYSPDB= ***** GX
SS$WRG= 000000
SS$YSZ= 007600
S.ALFL 000044
S.BLKC 000033
S.COST 000001
S.COUB 000044
S.COUL= 000040
S.CTL 000006
S.DLCF 000102
S.ERR 000010
S.FLG 000000
S.FNCP 000004
S.HTNA 000023
S.LBE 000044
S.LEN 000004
S.LGR 000031
S.LGTH= 000050
S.LIN 000004
S.LINK 177776
S.LMA 000026
S.LMOS 000030
S.LMRT 000027
S.LMS 000025
S.LNK 000000
S.LSA 000005
S.NCUB 000000
S.NCUL= 000036
S.NKLR 000040
S.NKR 000037
S.NKRB 000052
S.NKRW 000045
S.NKSB 000051
S.NKSW 000046
S.NMST 000002
S.NTD 000024
S.OWNR 000003
S.PLA 000010
S.PLC 000022
S.PLL 000006
S.PLS 000011
S.PSA 000003
S.RBE 000042
S.RCB 000020
S.RCV 000014
S.RCVB 000072
S.RVCV 000056
S.RDE 000034
S.REPR 000050
S.REPS 000047
S.RST 000036
S.RTEC 000042
S.RTY 000001
S.SELC 000066

```

F 9

MACRO NAME REFERENCES

ANBDFS	#5-57	5-75							
ASL\$	#5-59								
ASR\$	#5-57								
CALL	8-339	8-341	8-342	8-343	9-383	9-385	9-400	9-402	9-413
	9-425	10-455	10-457	10-461	10-479	10-481	10-497	10-510	10-526
	10-535	10-538	11-571	11-573	11-593	11-609	11-621	11-629	12-664
	12-678	12-680	12-682	12-684	12-698	12-709	12-714	12-718	12-725
	12-737	12-742	12-750	13-839	14-864	15-889	15-891	15-895	15-899
	15-907	15-912	15-921	15-928	15-930	16-954	17-991	17-996	18-1023
	18-1061	18-1071	19-1095	19-1099	19-1112	19-1132	19-1136	19-1142	19-1154
	19-1168	19-1170	21-1236	21-1245	21-1254	21-1263	22-1295	22-1298	22-1307
	23-1349	25-1397	25-1407	25-1408	26-1434	26-1436	26-1439	26-1456	26-1460
	26-1475	26-1476	26-1484	26-1487	26-1491	26-1511	26-1524	26-1541	26-1556
	26-1562	26-1565	26-1569	26-1574	26-1577	26-1583	26-1594	26-1599	26-1600
	26-1613	26-1627	26-1630	26-1643	26-1646	26-1658	26-1664	26-1673	26-1684
	26-1695	26-1700	26-1706	26-1710	26-1713	26-1735	26-1749	26-1754	26-1759
	26-1763	26-1766	26-1769	26-1775	26-1781	26-1791	26-1797	26-1798	26-1810
	26-1815	26-1819	26-1821	27-1923	27-1925	27-1941	27-1949	27-1955	27-1957
	27-1985	27-1995	27-2003	27-2014	27-2016	27-2064	27-2073	27-2083	27-2091
	27-2099	27-2112	27-2121	27-2131	28-2155	28-2157	28-2173	28-2180	28-2190
	28-2220	28-2224	28-2234	29-2258	29-2260	29-2272	29-2275	30-2310	30-2313
	31-2330	31-2342	31-2347	31-2352	31-2354	31-2358	31-2361	31-2368	31-2370
	33-2430	33-2439	33-2449	33-2456	33-2464	33-2467	33-2475	33-2477	33-2484
	34-2527	34-2529	34-2534	34-2539	34-2553	34-2570	34-2574	34-2581	34-2589
	35-2635	35-2636	35-2641	35-2642	35-2663	35-2665	35-2671	35-2673	36-2707
	37-2743	37-2754	37-2756	37-2766	37-2767	37-2771	37-2773	37-2780	37-2782
	39-2850	39-2852	39-2857	39-2864	39-2880	39-2894	39-2896	40-2926	40-2927
	40-2951	40-2966	40-2969	40-2977	40-2989	40-2991	40-3001	40-3005	40-3014
	40-3018	40-3023	40-3024	40-3046	40-3053	40-3056	40-3063	40-3065	41-3147
	42-3171	42-3176	42-3179	42-3190	42-3183	44-3279	44-3280	44-3296	45-3363
	46-3390	48-3454	48-3459	48-3482	48-3484	48-3501	48-3504	49-3557	49-3568
	49-3578	49-3583	49-3645	49-3648	50-3696	50-3702	50-3714	51-3756	51-3783
	51-3793	51-3796	51-3801	51-3806	51-3811	51-3814	52-3877	52-3892	52-3907
	53-3952	53-3959	53-3960	53-4006	53-4021	53-4025	53-4035	53-4049	53-4053
	53-4080	53-4081	53-4093	53-4095	53-4119	53-4127	53-4208	53-4212	53-4224
	53-4258	53-4262	53-4271	53-4273	54-4333	54-4336	54-4352	54-4353	54-4355
	55-4404	55-4405	55-4447	55-4454	55-4456	55-4473	55-4492	55-4517	55-4519
	56-4587	56-4597	57-4628	57-4639	57-4649	57-4662	58-4682	59-4716	59-4733
	59-4747	59-4749	60-4778	60-4785	62-4850	62-4853	63-4881	63-4886	63-4897
	66-5002	66-5003							
CEACCS	#5-58	22-1307	25-1407	37-2766	42-3179	44-3279	45-3363	59-4747	
CTRDFS	#5-59	5-76							
CUGDFS	#5-59	5-77							
DCBDFS	#5-58	5-78							
DDCDF\$	#5-58	5-79							
DHBD\$	#5-60	5-102							
DIR\$	#5-58	8-344	9-424	9-427	10-534	10-537	11-627	11-631	12-752
		26-1823	27-2127	28-2230	28-2233	43-3228			26-1564
DMPDFS	#5-59	5-80							
DSTD\$	#5-59	5-81							
DTED\$	#5-59	5-82							
ECDD\$	#5-60	5-83							

213		.SBTTL	ERROR MESSAGES
214		:	
215		:	ERROR CODES
216		:	
217	000300	ERMSG\$	<XPT channel not in data base>
218	000334	ERBLK\$	ERR1
219	000340	ERMSG\$	<Line not assigned to XPT>
220	000370	ERBLK\$	ERR2
221	000374	ERMSG\$	<XPT proress not loaded>
222	000422	ERBLK\$	ERR3
223	000426	ERMSG\$	<XPT data base not allocated>
224	000461	ERBLK\$	ERR4
225	000464	ERMSG\$	<XPT channel table inconsistent>
226	000522	ERBLK\$	ERR5
227	000526	ERMSG\$	<NETACP cannot be mounted>
228	000536	ERBLK\$	ERR6
229	000562	ERMSG\$	<Lines must be loaded before they can be enabled>
230	000641	ERBLK\$	ERR7
231	000644	ERMSG\$	<Line not loaded>
232	000663	ERBLK\$	ERR77
233	000666	ERMSG\$	<NETACP not installed>
234	000712	ERBLK\$	ERR8
235	000716	ERMSG\$	<RSX11M system pool allocation failure>
236	000763	ERBLK\$	ERR9
237	000766	ERMSG\$	<NETACP must be 'fixed' in RSX11S systems>
238	001036	ERBLK\$	ERR10
239	001042	ERMSG\$	<Network initializer not installed>
240	001103	ERBLK\$	ERR12
241	001106	ERMSG\$	<PLB allocation failure>
242	001134	ERBLK\$	ERR13
243	001140	ERMSG\$	<Network initializer not fixed>
244	001175	ERBLK\$	ERR14
245	001200	ERMSG\$	<Wrong network initializer installed>
246	001243	ERBLK\$	ERR15
247	001246	ERMSG\$	<RSX11M system pool allocation failure>
248	001313	ERBLK\$	ERR16
249	001316	ERMSG\$	<Invalid node option>
250	001341	ERBLK\$	ERR17
251	001344	ERMSG\$	<Object!Remote block allocation failure>
252	001412	ERBLK\$	ERR18
253	001416	ERMSG\$	<Node not in system>
254	001440	ERBLK\$	ERR19
255	001444	ERMSG\$	<Object not in system>
256	001470	ERBLK\$	ERR20
257	001474	ERMSG\$	<Line not in system>
258	001516	ERBLK\$	ERR21
259	001522	ERMSG\$	<SLT and reverse mapping table mismatch>
260	001570	ERBLK\$	ERR22
261	001574	ERMSG\$	<Access Verification not supported>
262	001635	ERBLK\$	ERR23
263	001640	ERMSG\$	<Alias not in system>
264	001663	ERBLK\$	ERR24
265	001666	ERMSG\$	<Alias block allocation failure>
266	001724	ERBLK\$	ERR25
267	001730	ERMSG\$	<Parameter missing, Scope>
268	001760	ERBLK\$	ERR26
269	001764	ERMSG\$	<Event logging not supported>

E 12

```

758          .SBTTL $VCOST - SET LINE COST
759          +
760          **-$VCOST - SET LINE COST
761          :
762          : INPUTS:
763          :   $SLTA = SLT ADDRESS
764          :   RQB = REQUEST BLOCK
765          :
766          : OUTPUTS:
767          :   C-BIT = SUCCESS/FAILURE
768          :
769          : -
770
771          $VCOST::
772          002006 016700 000000G      MOV    $SLTA,R0      ; GET THE SLT ADDRESS
773          002012 032760 000000G 000000G  BIT    #LF.RDY,L.FLG(R0) ; IS THE LINE READY ?
774          002020 001436          BEQ    101$          ; IF EQ, NO .. ERROR
775          002022 005001          CLR    R1            ; INIT REG
776          002024 032760 000000G 000000G  BIT    #LF.BRO,L.FLG(R0) ; ETHERNET DEVICE ?
777          002032 001003          BNE    5$           ; IF YES - BRANCH
778          002034 116001 000000G          MOVB   L.NSTA(R0),R1 ; ELSE, GET THE NUMBER OF TRIBUTARIES
779          002040 001010          BNE    10$          ; IF NE, LINE IS MULTIPOINT
780          :
781          : PT-TO-PT LINE
782          :
783          002042 116760 000020G 000000G 5$: MOVB   RQB+LO.COS,L.COST(R0) ; SET NEW LINE COST
784          002050 032760 000000G 000000G  BIT    #LF.BRO,L.FLG(R0) ; BROADCAST LINE ?
785          002056 001011          BNE    15$          ; IF YES - BRANCH
786          002060 000415          BR     20$          ; AND LEAVE
787          :
788          : MULTIPOINT LINE
789          :
790          002062 020167 000000G 10$:  CMP    R1,$TRIB      ; IS THE TRIBUTARY ON THIS LINE ?
791          002066 101417          BLOS   102$          ; IF LOS, NO .. ERROR
792          002070 016701 000000G          MOV    $TRIB,R1    ; ELSE, GET TRIBUTARY NUMBER
793          002074 006301          ASL    R1              ; CONVERT TO A DOUBLE WORD INDEX
794          002076 006301          ASL    R1              ;
795          002100 060001          ADD     R0,R1            ; POINT INTO SLT
796          002102 062701 000000G 15$:  ADD     #L.MPF,R1    ; POINT TO STATION INFO IN SLT
797          002106 116761 000020G 000000G  MOVB   RQB+LO.COS,S.COST(R1) ; AND SET NEW COST
798          002114          20$:  RETURN
799          :
800          : ERROR CONDITIONS
801          :
802          002116 101$:  ERRPT$ ERR35,$EROUT ; LINE IN WRONG STATE
803          002126 102$:  ERRPT$ ERR21,$EROUT ; LINE NOT IN SYSTEM

```

F 12


```

1308 004074 116767 000060G 000042G      MOVB    LO.INC+RQB,$BFR+D$INCT ; STORE INCOMING TIMER VALUE
1309 004102                                     PUTRC    ; WRITE HOME BLOCK
1310
1311 004106                                     RETURN
1312
1313
1314      ;+
1315      ; OUTGOING TIMER
1316      ; INPUTS:
1317      ; RQB+LO.OUT = OUTGOING TIMER VALUE
1318      ;
1319      ; OUTPUTS:
1320      ; DECNET HOME BLOCK UPDATED
1321      ; -
1322
1323
1324 004110                                     $VOUT:: GETRV  $DECT,#D$END ; READ HOME BLOCK
1325 004130 116767 000056G 000043G      MOVB    LO.OUT+RQB,$BFR+D$OUTT ; STORE OUTGOING TIMER VALUE
1326 004136                                     PUTRC    ; WRITE HOME BLOCK
1327
1328 004142                                     RETURN
1329
1330
1331      ;+
1332      ; INACTIVITY TIMER VALUE
1333      ; INPUTS:
1334      ; RQB+LO.INA = INACTIVITY TIMER VALUE
1335      ;
1336      ; OUTPUTS:
1337      ; DECNET HOME BLOCK UPDATED
1338      ; -
1339
1340
1341 004144                                     $VINA:: GETRV  $DECT,#D$END ; READ HOME BLOCK
1342 004164 116767 000062G 000044G      MOVB    LO.INA+RQB,$BFR+D$INAC ; STORE INACTIVITY TIMER VALUE
1343 004172                                     PUTRC    ; WRITE HOME BLOCK
1344
1345 004176                                     RETURN
1346
1347
1348      ;+
1349      ; RETRANSMIT FACTOR
1350      ; INPUTS:
1351      ; RQB+LO.RET = RETRANSMIT FACTOR
1352      ;
1353      ; OUTPUTS:
1354      ; DECNET HOME BLOCK UPDATED
1355      ; -
1356
1357
1358 004200                                     $VRET:: GETRV  $DECT,#D$END ; READ HOME BLOCK
1359 004220 116767 000064G 000050G      MOVB    LO.RET+RQB,$BFR+D$RETF ; STORE RETRANSMIT FACTOR
1360 004226                                     PUTRC    ; WRITE HOME BLOCK
1361
1362 004232                                     RETURN
1363
1364

```

```

1816                                     .SBTTL $RPASW - SET/CLEAR RECEIVE PASSWORD
1817                                     +
1818                                     *** - $RPASW - SET/CLEAR RECEIVE PASSWORD
1819                                     :
1820                                     INPUTS:
1821                                     $CMAND = SET/CLEAR
1822                                     :
1823                                     OUTPUTS:
1824                                     APPROPRIATE ACTION TAKEN
1825                                     :
1826                                     -
1827 $RPASW::
1828 CALL CHKXPT                                     : BE SURE XPT IS LOADED
1829 BCS 40$                                         : IF CS, XPT NOT LOADED
1830 GETRV R2,#N$XLEN                               : READ IN POINTERS
1831 MOV N$VER+2+$BFR,R2                           : ADDRESS OF TRANSMIT/RECEIVE PASSWORDS
1832 GETRV R2,#18.                                  : FLAG AND PASSWORDS
1833
1834 006176 032767 000000G 000000G               BIT #CM$SET,$CMAND           : SET PASSWORD ?
1835 006204 001415                                     BEQ 20$                           : NO
1836
1837                                     : SET PASSWORD
1838                                     :
1839 006206 012704 000002G 000000G               MOV #V$RCV+$BFR,R4               : SET DESTINATION ADDRESS
1840 006212 052767 100000 000000G               BIS #V$RCV,V$FLG+$BFR           : SET FLAG INDICATING VALID RECEIVE PASSWORD
1841 006220 012705 000030G                       MOV #RQB+LO.RPA,R5              : POINT AT PASSWORD SOURCE
1842 006224 012703 000010                       MOV #8,R3                     : NUMBER OF CHARACTERS TO MOVE
1843 006230 112524 10$: MOVB (R5)+,(R4)+         : MOVE STRING
1844 006232 005303                               DEC R3
1845 006234 001375                               BNE 10$
1846 006236 000405                               BR 30$
1847
1848                                     : CLEAR PASSWORD
1849                                     :
1850 006240 042767 100000 000000G 20$: BIC #V$RCV,V$FLG+$BFR           : CLEAR RECEIVE FLAG
1851 006246 005067 000002G                       CLR V$RCV+$BFR                 : FORCE ILLEGAL PASSWORD
1852
1853 006252 30$: PUTRC                               : REWRITE DATA BLOCK
1854 006256 40$: RETURN

```

```

2452                                     .SBTTL VALI - REMOVE ALL OR KNOWN ALIASES
2453
2454                                     ;+
2455                                     ;**VALI-REMOVE ALL OR KNOWN ALIASES
2456                                     ;
2457                                     ; INPUTS:
2458                                     ; $QUAL = QUALIFIERS
2459                                     ;
2460                                     ; OUTPUTS:
2461                                     ; NONE.
2462                                     ;
2463                                     ; -
2464 VALI:
2465 011734 032767 000004 000000G BIT #Q$KNO,$QUAL ; IS THIS FOR KNOWN ALIASES ?
2466 011734 001410 BEQ 10$ ; IF EQ, NO .. OKAY
2467 011744 032767 000001 000000G BIT #Q$SCO,$OPTON ; WAS THE SCOPE GIVEN ?
2468 011752 001004 BNE 10$ ; IF NE, YES .. OKAY
2469 011754 ERRPT$ ERR26,$EROUT ; ELSE, SCOPE PARAMETER MISSING
2470
2471 011764 103403 10$: CALL $FNALI ; FIND THE NEXT ALIAS
2472 011770 BCS 20$ ; IF CS, NO MORE
2473 011772 CALL REMALI ; AND REMOVE THE ALIAS
2474 011776 BR 10$ ; GET NEXT ALIAS
2475 012000 20$: RETURN

```

```

2955 .SBTTL CKMDEV - CHECK FOR VALID RSX11M DEVICE NAME
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974 013634 012702 000002 CKMDEV: MOV #2,R2 ; DEVICE NAME IS 2 ASCII CHARACTERS
2975 013640 112001 10$: MOV (R0)+,R1 ; GET ASCII CHARACTER
2976 013642 022701 000132 CMP #'Z',R1 ; IS IT AN ALPHABETIC?
2977 013646 103435 BLO 30$ ; BR IF NO
2978 013650 122701 000101 CMPB #'A',R1 ;
2979 013654 101032 BHI 20$ ; BR IF NOT ALPHABETIC
2980 013656 101024 MOV (R4)+,R1 ; STORE CHARACTER
2981 013660 005302 DEC R2 ; MORE TO CHECK?
2982 013662 003366 BGT 10$ ; BR IF YES
2983 013664 005001 CLR R1 ; ASSUME NO UNIT NUMBER SPECIFIED
2984 013666 122027 000072 CMPB (R0)+, #' ': ; ANY UNIT NUMBER SPECIFIED?
2985 013672 001411 BEQ 20$ ; BR IF NO
2986 013674 005300 DEC R0 ; POINT TO UNIT NUMBER
2987 013676 CALL $COTB ; CONVERT UNIT NUMBER TO BINARY
2988 013702 022702 000072 CMP #' ',R2 ; VALID TERMINATOR?
2989 013706 001015 BNE 30$ ; BR IF NO
2990 013710 020127 000377 CMP R1,#MXUNT ; LEGAL UNIT NUMBER?
2991 013714 101012 30$: BHI 30$ ; BR IF NO
2992 013716 010124 MOV R1,(R4) ; STORE UNIT NUMBER
2993 013720 010046 MOV R0,-(SP) ; SAVE R0
2994 013722 016400 177774 MOV -4(R4),R0 ; GET DEVICE NAME
2995 013726 CALL FNDDCB ; IS DEVICE IN SYSTEM?
2996 013732 012600 MOV (SP)+,R0 ; RESTORE R0
2997 013734 103403 BCS 40$ ; BR IF NO
2998 013736 000241 CLC ; INDICATE VALID DEVICE NAME
2999 013740 000401 BR 40$ ; AND EXIT
3000 013742 000261 30$: SEC ; INDICATE INVALID DEVICE NAME
3001 013744 40$: RETURN

```

```

1272      .SBTTL BLDNOD - BUILD NODE ENTITY
1273
1274      +
1275      BLDNOD - BUILD NODE ENTITY
1276
1277      INPUTS:
1278          R0 - POINTER TO NEXT FREE BYTE IN OUTPUT BUFFER
1279          R3 - ADDRESS OF FILTER BLOCK
1280
1281      OUTPUTS:
1282          R0 - POINTER TO NEXT FREE BYTE IN OUTPUT BUFFER
1283
1284      -
1285
1286      BLDNOD: MOVB #''' (R0)+ ; STORE DELIMITER
1287              MOV #NSPEC,R1 ; POINT TO ENTITY STRING
1288              MOVB (R1)+,(R0)+ ; STORE STRING
1289              BNE 5$ ; ...
1290              TSTB -(R0) ; BACK UP OVER SPACE
1291              MOVB #'',(R0)+ ; STORE SPACE IN OUTPUT BUFFER
1292              MOV F,ADD(R3),R1 ; GET NODE ADDRESS
1293              CLR R2 ; SUPPRESS LEADING ZEROES
1294              CALL $CBDMG ; CONVERT IT TO ASCII
1295
1296
1297              SWTCHI #$HMBUF ; SET UP INPUT BUFFER
1298              GETRV $DECP,$DSEND ; READ DECNET HOME BLOCK
1299              MOV #$HMBUF+$SRNN+2,R2 ; GET REMOTE LIST ADDRESS
1300
1301              SWTCHI #TEMP4 ; USE TEMPORARY BUFFER
1302
1303              MOV (R2),TEMP4 ; SAVE ADDRESS OF NEXT
1304              MOV R2,R1 ; SAVE ADDRESS OF PREVIOUS
1305              MOV TEMP4,R2 ; GET NEXT REMOTE IN LIST
1306              BEQ 60$ ; BR IF END OF LIST
1307              CEACCS R2 ; CONVERT TO MAPPED ADDRESS
1308              GETAD R2,$R,LEN ; READ REMOTE IN INTERNAL BUFFER
1309              CMP F,ADD(R3),TEMP4+R,ADD ; IS THIS THE RIGHT REMOTE?
1310              BNE 10$ ; BR IF NO
1311              MOVB #SPACE,(R0)+ ; STORE SPACE
1312              MOVB #'',(R0)+ ; STORE LEFT PAREN
1313              MOV #6,R1 ; MAXIMUM LENGTH OF NODE NAME
1314              MOV #TEMP4+R,NAM,R2 ; POINT TO NODE NAME
1315              MOVB (R2)+,(R0) ; STORE NODE NAME
1316              CMPB (R0)+,$SPACE ; END OF NODE NAME?
1317              BEQ 40$ ; BR IF YES
1318              DEC R1 ; DECREMENT COUNT
1319              BGT 30$ ; BR IF MORE TO STORE
1320              BR 50$ ; CONTINUE
1321              TSTB -(R0) ; BACK UP OVER SPACE
1322              MOVB #'',(R0)+ ; STORE RIGHT PAREN
1323              SWTCHI ; RESTORE INPUT BUFFER TO DEFAULT ($BFR)
1324              MOVB #SPACE,(R0)+ ; STORE TRAILING DELIMITER
1325              MOVB #SPACE,(R0)+ ; STORE SPACES
1326              RETURN ; ...
1327
10$:      MOV (R2),TEMP4 ; SAVE ADDRESS OF NEXT
11$:      MOV R2,R1 ; SAVE ADDRESS OF PREVIOUS
12$:      MOV TEMP4,R2 ; GET NEXT REMOTE IN LIST
13$:      BEQ 60$ ; BR IF END OF LIST
14$:      CEACCS R2 ; CONVERT TO MAPPED ADDRESS
15$:      GETAD R2,$R,LEN ; READ REMOTE IN INTERNAL BUFFER
16$:      CMP F,ADD(R3),TEMP4+R,ADD ; IS THIS THE RIGHT REMOTE?
17$:      BNE 10$ ; BR IF NO
18$:      MOVB #SPACE,(R0)+ ; STORE SPACE
19$:      MOVB #'',(R0)+ ; STORE LEFT PAREN
20$:      MOV #6,R1 ; MAXIMUM LENGTH OF NODE NAME
21$:      MOV #TEMP4+R,NAM,R2 ; POINT TO NODE NAME
22$:      MOVB (R2)+,(R0) ; STORE NODE NAME
23$:      CMPB (R0)+,$SPACE ; END OF NODE NAME?
24$:      BEQ 40$ ; BR IF YES
25$:      DEC R1 ; DECREMENT COUNT
26$:      BGT 30$ ; BR IF MORE TO STORE
27$:      BR 50$ ; CONTINUE
28$:      TSTB -(R0) ; BACK UP OVER SPACE
29$:      MOVB #'',(R0)+ ; STORE RIGHT PAREN
30$:      SWTCHI ; RESTORE INPUT BUFFER TO DEFAULT ($BFR)
31$:      MOVB #SPACE,(R0)+ ; STORE TRAILING DELIMITER
32$:      MOVB #SPACE,(R0)+ ; STORE SPACES
33$:      RETURN ; ...

```

```

1907      .SBTT' $DCIR - SHOW CIRCUIT
1908
1909      ;+
1910      ; $DCIR - SHOW KNOWN CIRCUITS
1911      ; SHOW ACTIVE CIRCUITS
1912      ; SHOW CIRCUIT X
1913
1914      ; INPUTS: NONE
1915
1916      ; OUTPUTS: THE CIRCUIT INFORMATION IS DISPLAYED ON THE USER'S TERMINAL
1917
1918      ; ALL REGISTERS ARE DESTROYED
1919
1920
1921
1922 $DCIR:: VNPDBG DCIR      ; VNP DEBUGGING BREAK POINT      ;[TD001]
1923      CALL CKTYPE      ; GET COMMAND TYPE      ;[TD001]
1924      BCC 10$           ; BR IF SUCCESS      ;***-2
1925      ERRPT$ SYNERR,$ERDUT      ; ELSE SYNTAX ERROR
1926 10$: MOV $SLTMA,R5      ; GET ADDRESS OF SLT ADDRESS TABLE
1927      MOV $SLTNM,R3      ; GET NUMBER OF SLT'S IN SYSTEM
1928      MOV #TEMP2,R1      ; STORAGE FOR HEADER MESSAGE
1929      TSTB $QUAL          ; SHOW SPECIFIC CIRCUIT?
1930      BEQ 140$           ; BR IF YES
1931      CMPB #QF$KND,$QUAL      ; SHOW KNOWN CIRCUITS?
1932      BNE 20$           ; BR IF NO
1933      JMP 300$          ; BR IF YES
1934
1935      ; SHOW ACTIVE CIRCUITS
1936
1937 20$: MOV #CACTIV,R0      ; MUST BE SHOW ACTIVE CIRCUITS
1938      MOVB (R0)+,(R1)+      ; STORE THE HEADER MESSAGE
1939      BNE 30$           ;
1940      CLR R4             ; INDICATE HEADER TO BE DISPLAYED
1941      CALL NXCIR          ; GET NEXT SYSTEM LINE TABLE IN SYSTEM
1942      BCC 80$           ; BR IF FOUND ONE
1943
1944      ; DISPLAY PSI PERMANENT CIRCUITS
1945
1946      MOV $PSIPT,R0      ; IS THIS A PSI SYSTEM?
1947      BEQ 70$           ; BRANCH IF NO - DONE
1948      ADD #H$PVC,R0      ; GET ADDRESS OF CIRCUIT LISTHEAD
1949      GETRV R0,#2        ; READ IN LISTHEAD
1950      CLR R0             ; NO SEARCH CRITERION (ALL CIRCUITS)
1951      MOV #C$LEN,R1      ; LENGTH OF PVC BLOCK IN OLD FORMAT
1952      BIT #PF.XDF,$PFLAG ; CHECK FOR EXTENDED FORMAT
1953      BEQ 60$           ; IF NOT EXTENDED, THESE PARAMETERS OK
1954      ADD #MAXID,R1      ; OTHERWISE, PVC BLOCK IS LONGER
1955      CALL FDNBLK        ; LOOK FOR A CIRCUIT
1956      BCS 70$           ; BRANCH IF END OF LIST
1957      CALL DISPVC        ; DISPLAY PVC INFO
1958      INC R4             ; UPDATE HEADER DISPLAY FLAG
1959      BR 50$            ; GET NEXT PVC NAME BLOCK
1960
1961 50$: JMP 390$           ; BR IF NO MORE
1962 60$: BIT #LF.BRO,L.FLG(R0) ; ETHERNET LINE ?
1963      BNE 90$           ; IF YES - DO NOT HANDLE AS MPT

```

```

2470 017034 012700 000064G      MOV    #SBFR+R$ACC,R0      ; Copy account to message      ;[TD001]
2471 017040 012702 001034'      MOV    #TEMP1,R2      ;[TD001]
2472 017044 112022              75$:  MOV    (R0)+(R2)+      ;[TD001]
2473 017046              SOB    R1,75$      ;[TD001]
2474 017052 105012              CLRB   (R2)      ;[TD001]
2475 017054              ERRPT$  ACNS5,MSGOUT      ; Display account      ;[TD001]
2476              ;[TD001]
2477 017064              80$:  ERRPT$  ACNS7,MSGOUT      ; Blank line after each block      ;[TD001]
2478 017074 000651              BR     10$      ; Repeat      ;[TD001]
2479              ;[TD001]
2480 017076 032767 000004 162470 90$:  BIT    #F.FND,FLAGS      ; Check if anything displayed      ;[TD001]
2481 017104 001015              BNE    100$      ; Branch if message printed      ;[TD001]
2482 017106 032767 004000 000000G    BIT    #OP$NET,$O$PTON      ; Check if specific network wanted      ;[TD001]
2483 017114 001005              BNE    95$      ; branch if specific network      ;[TD001]
2484 017116              92$:  ERRPT$  NOINFO,MSGOUT      ; Indicate no information      ;[TD001]
2485 017126 000404              BR     100$      ;[TD001]
2486 017130              95$:  ERRPT$  ACNS6,MSGOUT      ; Indicate network not found      ;[TD001]
2487 017140              100$:  RETURN      ;[TD001]
2488              ;[TD001]
2489              ; Messages      ;[TD001]
2490              ;[TD001]
2491              ;[TD001]
2492 017142              ERMSG$ < Network = %A>      ;[TD001]
2493 017157              ERBLK$  ACNS1,<TEMP1>      ;[TD001]
2494              ;[TD001]
2495 017164              ERMSG$ < Node = %A>      ;[TD001]
2496 017201              ERBLK$  ACNS2,<TEMP1>      ;[TD001]
2497              ;[TD001]
2498 017206              ERMSG$ < User = %A>      ;[TD001]
2499 017223              ERBLK$  ACNS3,<TEMP1>      ;[TD001]
2500              ;[TD001]
2501 017230              ERMSG$ < Password set>      ;[TD001]
2502 017250              ERBLK$  ACNS4      ;[TD001]
2503              ;[TD001]
2504 017254              ERMSG$ < Account = %A>      ;[TD001]
2505 017274              ERBLK$  ACNS5,<TEMP1>      ;[TD001]
2506              ;[TD001]
2507 017302              ERMSG$ < Remote network not is system%N>      ;[TD001]
2508 017344              ERBLK$  ACNS6      ;[TD001]
2509              ;[TD001]
2510 017350              ERMSG$ < >      ;[TD001]
2511 017351              ERBLK$  ACNS7      ;[TD001]

```

```

3023 023072          CALL   BCDASC          ; Convert to ASCII (R1 already setup) ;[TD001]
3024 023076          ERRPT$ DSMS4,MSGOUT    ; Display DTE address          ;[TD001]
3025
3026          ; Display CUG name                ;[TD001]
3027
3028 023106 032767 000000G 000000G 60$: BIT    #PF.XDF,$PFLAG    ; Check for extended data format ;[TD001]
3029 023114 001411          BEQ    65$          ; Branch if not extended          ;[TD001]
3030 023116 012700          MOV    #$BFR+$D$GVBL,R0              ; Get start address of variable data ;[TD001]
3031 023122 116701 000022G          MOV    $BFR+$D$SL,R1          ; Skip destination name            ;[TD001]
3032 023126 060100          ADD    R1,R0          ; Now pointing at CUG name          ;[TD001]
3033 023130 116701 000023G          MOV    $BFR+$D$UGL,R1          ; Get length of CUG name            ;[TD001]
3034 023134 001423          BEQ    70$          ; Branch if no CUG name            ;[TD001]
3035 023136 000410          BR      68$          ; Join common code to display      ;[TD001]
3036
3037 023140 122767 000040 000022G 65$: CMP    #SPACE,$D$CUGN+$BFR    ; Is there a CUG name?              ;[TD001]
3038 023146 001416          BEQ    70$          ; Branch if no                      ;[TD001]
3039 023150 012700          MOV    #$BFR+$C$CUGN,R0              ; Address CUG name                  ;[TD001]
3040 023154 012701 000006          MOV    #6,R1          ; Get length of CUG name            ;[TD001]
3041
3042 023160 012702 001034' 68$: MOV    #TEMP1,R2          ; Address output buffer              ;[TD001]
3043 023164 112022          69$: MOV    (R0)+,(R2)+          ; Create output string              ;[TD001]
3044 023166          SOB    R1,69$          ;[TD001]
3045 023172 105012          CLRB   (R2)          ;[TD001]
3046 023174          ERRPT$ DSMS5,MSGOUT    ; Display CUG name                  ;[TD001]
3047
3048          ; Display subaddress range            ;[TD001]
3049
3050 023204 016767 000016G 155622 70$: MOV    D$SLO+$BFR,TEMP1    ; Store range beginning            ;[TD001]
3051 023212 026767 000016G 000020G          CMP    D$SLO+$BFR,D$SHI+$BFR    ; Range beginning = range end?      ;[TD001]
3052 023220 001006          BNE    80$          ; Branch if no                      ;[TD001]
3053 023222          ERRPT$ DSMS6,MSGOUT    ; Display subaddress range          ;[TD001]
3054 023232 000167 177126          JMP     10$          ; and continue                     ;[TD001]
3055 023236 016767 000020G 155616 80$: MOV    D$SHI+$BFR,TEMP2    ; Get range end                    ;[TD001]
3056 023244          ERRPT$ DSMS7,MSGOUT    ; Display subaddress range          ;[TD001]
3057 023254 000167 177104          JMP     10$          ; Continue                         ;[TD001]
3058
3059 023260 132767 000004 156306 90$: BIT    #F.FND,FLAGS    ; Found a destination block?        ;[TD001]
3060 023266 001015          BNE    110$          ; Branch if found                  ;[TD001]
3061 023270 032767 000200 000000G          BIT    #OP$DST,$OPTON    ; Show specific destination?        ;[TD001]
3062 023276 001005          BNE    100$          ; Branch if yes                    ;[TD001]
3063 023300          ERRPT$ NOINFO,MSGOUT    ; Display NO INFORMATION message    ;[TD001]
3064 023310 000404          BR      110$          ; and exit                        ;[TD001]
3065 023312          100$: ERRPT$ DSMS8,MSGOUT    ; DESTINATION NOT FOUND            ;[TD001]
3066 023322          110$: RETURN          ;[TD001]
3067
3068          ; MESSAGES                          ;[TD001]
3069
3070 023324          ERMSG$ < Name = %A>
3071 023336          ERBLK$ DSMS1,<TEMP1>
3072
3073 023344          ERMSG$ < Priority = %D, Object = %R%R>
3074 023404          ERBLK$ CSMS2A,<TEMP1,TEMP2,TEMP2+2>
3075
3076 023416          ERMSG$ < Priority = %D, Object = %G>
3077 023454          ERBLK$ DSMS2B,<TEMP1,TEMP2>
3078
3079 023464          ERMSG$ < Priority = %D, Object = %A> ; New format objects ;[TD001]

```



```

3590      ; GET DESTINATION INFORMATION
3591      ;
3592 026102 012702 001062' 40$: MOV #TEMP2,R2      ; TEMPORARY BUFFER FOR DESTINATION INFO
3593 026106 010300      MOV R3,R0      ; POINT TO ALIAS NAME BLOCK
3594 026110 062700 000012 ADD #A.REM,R0      ; POINT TO REMOTE NODE NAME
3595      000003      .REPT 3
3596      MOV (R0)+,(R2)+      ; SAVE NAME
3597      .ENDR
3598 026122 122742 000040 50$: CMPB #SPACE,-(R2)      ; DELETE TRAILING BLANKS
3599 026126 001775      BEQ 50$
3600 026130 005202      INC R2      ; POINT PAST NON-BLANK
3601      ;
3602      ; UIC
3603      ;
3604 026132 010301      MOV R3,R1      ; ADDRESS OF ALIAS NAME BLOCK
3605 026134 062701 000020 ADD #A.ACC,R1      ; POINT TO DESTINATION INFO
3606 026140 112722 000057 MOVB #SLASH,(R2)+      ; INSERT SEPARATING CHARACTER
3607 026144 112100      MOVB (R1)+,R0      ; GET BYTE COUNT FOR UIC
3608 026146 001403      BEQ 65$      ; CHECK FOR PASSWORD
3609 026150 112122      MOVB (R1)+,(R2)+      ; STORE UIC
3610 026152 005300      DEC R0      ; MORE TO COPY?
3611 026154 003375      BGT 60$      ; BR IF YES
3612      ;
3613      ; PASSWORD
3614      ;
3615 026156 112722 000057 65$: MOVB #SLASH,(R2)+      ; INSERT SEPARATING CHARACTER
3616 026162 112100      MOVB (R1)+,R0      ; GET BYTE COUNT FOR PASSWORD
3617 026164 001407      BEQ 80$      ; BR IF NONE SPECIFIED
3618 026166 060001      ADD R0,R1      ; POINT PAST PASSWORD
3619 026170 012700 000003 MOV #3,R0      ; INSERT
3620 026174 112722 000056 70$: MOVB #PERIOD,(R2)+      ; STORE PASSWORD
3621 026200 005300      DEC R0      ; MORE TO COPY?
3622 026202 003374      BGT 70$      ; BR IF YES
3623      ;
3624      ; ACCOUNT NUMBER
3625      ;
3626 026204 112100      MOVB (R1)+,R0      ; GET BYTE COUNT FOR ACCOUNT NUMBER
3627 026206 001405      BEQ 95$      ; BR IF NO ACCOUNT NUMBER
3628 026210 112722 000057 MOVB #SLASH,(R2)+      ; INSERT SEPARATING CHARACTER
3629 026214 112122      MOVB (R1)+,(R2)+      ; STORE ACCOUNT NUMBER
3630 026216 005300      DEC R0      ; MORE TO COPY?
3631 026220 003375      BGT 90$      ; BR IF YES
3632      ;
3633      ; CREATE ASCIZ STRING FOR DESTINATION INFORMATION
3634      ;
3635 026222 122742 000057 95$: CMPB #SLASH,-(R2)      ; TERMINATING CHARACTER A SLASH?
3636 026226 001003      BNE 98$      ; CREATE ASCIZ STRING
3637 026230 122742 000057 97$: CMPB #SLASH,-(R2)      ; ANOTHER SLASH?
3638 026234 001775      BEQ 97$      ; BR IF YES
3639 026236 112762 000000 000001 98$: MOVB #0,1(R2)      ; CREATE ASCIZ STRING
3640      ;
3641      ; DISPLAY DESTINATION INFORMATION
3642      ;
3643 026244 105767 152564 100$: TSTB TEMP1      ; WAS SCOPE SPECIFIED AS GLOBAL?
3644 026250 002005      BGE 110$      ; BR IF NO
3645 026252      ERRPT$ AMSG3,MSGOUT      ; DISPLAY GLOBAL SCOPE AND DESTINATION
3646 026262 000404      BR 120$      ; FINISH UP

```

```

4152 031332 012704 000064' 340$: MOV #DN,R4 : ELSE IT IS OFF
4153 031336 112425 350$: MOV (R4)+,(R5)+ : STORE THE STATE
4154 031340 001376 BNE 350$ : ...
4155
4156
4157 ;
4158 ; DISPLAY CIRCUIT SERVICE INFO
4159 ;
4160 031342 012705 001062' MOV #TEMP2,R5 : GET BUFFER ADDRESS
4161 031346 012704 000477' MOV #ENABLE,R4 : ASSUME SERVICE ENABLED
4162 031352 032760 040000 000000 BIT #LF.RDY,L.FLG(R0) : LINE LOADED ?
4163 031360 001431 BEQ 410$ : IF NO - BRANCH
4164 031362 105760 000014 TSTB L.NSTA(R0) : MULTIPOINT ?
4165 031366 001005 BNE 360$ : IF YES - BRANCH
4166 031370 032760 000040 000000 BIT #LF.SER,L.FLG(R0) : SERVICE DISABLED ?
4167 031376 001420 BEQ 400$ : IF NO - BRANCH
4168 031400 000415 BR 390$ : CONTINUE
4169
4170 031402 032760 000400 000000 360$: BIT #LF.BRO,L.FLG(R0) : BROADCAST LINE ?
4171 031410 001404 BEQ 370$ : IF NO - BRANCH
4172 031412 010003 MOV R0,R3 : COPY SLT ADDRESS
4173 031414 062703 000022 ADD #L.MPF,R3 : CALCULATE STATION ADDRESS
4174 031420 000401 BR 380$ : CONTINUE
4175
4176 031422 011603 370$: MOV (SP),R3 : GET STATION ADDRESS
4177 031424 132763 000001 000000 380$: BITB #SF.SER,S.FLG(R3) : SERVICE DISABLED ?
4178 031432 001402 BEQ 400$ : IF NO - BRANCH
4179 031434 012704 000507' 390$: MOV #DISAB,R4 : STORE DISABLE STRING ADDRESS
4180 031440 112425 400$: MOV (R4)+,(R5)+ : COPY STRING
4181 031442 001376 BNE 400$
4182
4183 ;
4184 ; LOOPBACK NODE NAME
4185 ;
4186 031444 105760 000014 410$: TSTB L.NSTA(R0) : MULTIPOINT CIRCUIT?
4187 031450 001004 BNE 420$ : BR IF YES
4188 031452 032760 001000 000000 BIT #LF.LPB,L.FLG(R0) : IS THERE A LOOPBACK NAME?
4189 031460 000405 BR 430$ : ...
4190 031462 010204 420$: MOV R2,R4 : GET STATION NUMBER
4191 031464 011605 MOV (SP),R5 : GET MULTIPOINT FLAG TABLE ADDRESS
4192 031466 032763 000004 000000 BIT #SF.LPB,S.FLG(R5) : IS THERE A LOOPBACK NAME?
4193 031474 001465 430$: BEQ 470$ : BR IF NO - DISPLAY STATE
4194 031476 016705 000000G MOV $SLTNM,R5 : COMPUTE THE SYSTEM LINE NUMBER
4195 031502 166605 000004 SUB 4(SP),R5 : ...
4196 031506 005305 DEC R5 : ...
4197 031510 006305 ASL R5 : FIND CURRENT PDV & CHANNEL ASSIGNMENT
4198 031512 066705 000000G ADD $LLCTA,R5 : COMPUTE ADDR IN THE REVERSE CHANNEL
4199 031516 011504 MOV (R5),R4 : MAPPING TABLE & GET CONTENTS
4200 031520 100003 BPL 440$ : IF MJ, ADDR OF TRIB MAPPING TABLE
4201 031522 010205 MOV R2,R5 : GET TRIBUTARY NUMBER
4202 031524 060405 ADD R4,R5 : POINT INTO MAPPING TABLE/2
4203 031526 006305 ASL R5 : FORM REAL ADDRESS
4204 031530 011505 440$: MOV (R5),R5 : GET PDV AND CHANNEL NUMBER
4205 031532 110504 MOV R5,R4 : GET CHANNEL NUMBER
4206 031534 052704 100000 BIS #100000,R4 : INDICATE CHANNEL NUMBER
4207
4208 031540 GETRV $DECP,T,#D$END : READ DECNET HOME BLOCK

```

```

4698 .SBTTL GTSID - GET NODE STATE AND IDENTIFICATION
4699
4700
4701
4702
4703
4704
4705
4706
4707
4708
4709
4710
4711
4712 034630
4713 034632 012701 001034'
4714 034636 012700 000077'
4715 034642 012702 001024'
4716 034646
4717 034652 103431
4718 034654 032767 000000G 000000G
4719 034662 001404
4720
4721
4722 034664 032767 002000 000034G
4723 034672 001421
4724
4725
4726 034674 005767 000012G 5$:
4727 034700 001416
4728
4729
4730
4731
4732
4733 034702
4734
4735
4736
4737 034722 032767 010000 000006G
4738 034730 001402
4739 034732 012700 000064'
4740 034736 112021
4741 034740 001376
4742
4743 034742
4744 034762 012700 000000G
4745 034766 062700 000020
4746 034772 011001
4747 034774
4748
4749 035004
4750 035022 012700 000000G
4751 035026 012701 001062'
4752
4753 035032 012002
4754 035034 112021 20$:

;+
; GTSID - GET NODE STATE AND IDENTIFICATION
; INPUTS: NONE
; OUTPUTS:
;   TEMP1 = NODE STATE (ASCII)
;   TEMP2 = NODE IDENTIFICATION (ASCII)
;-

GTSID: SAVRG <R5> ; SAVE R5
      MOV #TEMP1,R1 ; STORAGE FOR STATE STRING
      MOV #OFF,R0 ; ASSUME STATE IS OFF
      MOV #NTINAM,R2 ; GET NTINIT NAME IN RAD50
      CALL TSKSCH ; GET TCB ADDRESS
      BCS 10$ ; TASK NOT INSTALLED
      BIT #VF.SS,$VFLAG ; S-SYSTEM ?
      BEQ 5$ ; NO

      .IF NDF R$$MPL ;[MP01]
      BIT #T2.FXD,T.ST2+$BFR ; IS TASK FIXED
      BEQ 10$ ; TASK NOT FIXED ;[MP01]
      .ENDC

      TST T.RCVL+$BFR ; ANY PACKETS QUEUED ?
      BEQ 10$ ; NO - STATE IS OFF

      .IF DF R$$MPL ;[MP01]
      GETAD #140000,$RCVLNG,T.RCVL+$BFR ; READ IN SECONDARY POOL ;[MP01]
      .IFF
      GETRV T.RCVL+$BFR,$RCVLNG ; READ IN PACKET ;[MP01]
      .ENDC

      BIT #LS.CXO,LR.STS+6+$BFR ; IS EXECUTOR STATE ON QUEUED?
      BEQ 10$ ; BR IF NO
      MOV #ON,R0 ; EXECUTOR STATE IS ON
      MOVB (R0)+,(R1)+ ; STORE THE STRING
      BNE 10$ ; ...

      GETRV $DECP,$DSEND ; READ DECNET HOME BLOCK
      MOV #BFR,R0 ; GET NODE ID ADDRESS
      ADD #DLID,R0
      MOV (R0),R1 ; GET UNMAPPED ADDRESS OF NODE ID
      CEACC$ R1 ; MAP TO NODE ID STRING

      GETAD R1,$34 ; READ ID STRING
      MOV #BFR,R0 ; GET BUFFER ADDRESS
      MOV #TEMP2,R1 ; GET OUTPUT BUFFER ADDRESS

      MOV (R0)+,R2 ; GET CHARACTER COUNT
      MOVB (R0)+,(R1)+ ; STORE THE IDENTIFICATION STRING
  
```

S.SELT 000053	S3.NEC= 000020	TIDET 001646R	T\$KMG= 000000	T.TOD 000070
S.STA 000000	S3.PPT= 020000	TR.ALF 000102	T\$SMIN= 000000	T.TRLG= 000076
S.STE 000002	S3.PTH= 004000	TR.DEV 000100	T.ACT 000031	T.TRLN 000005
S.STEC 000043	S3.RAL= 000010	TR.LEN= 000210	T.ACTL 000052	T.TRPN 000004
S.STLG 000104	S3.RES= 010000	TR.MST 000074	T.ASTL 000016	T.UCB 000026
S.STLN 000035	S3.RUB= 040000	TR.PRE 000076	T.ATT 000062	T.XASQ 000010
S.STPN 000034	S3.TAB= 000002	TR.TLZ 000072	T.BTTC 000074	T.XBTC 000024
S.TIMC 000054	S3.TME= 002000	TSKSGH 035052R	T.BUF 000006	T.XDAT 000030
S.TLZ 000012	S3.TSY= 000040	TS.BLK= 171700	T.CHA 000032	T2.ABO= 000100
S.TMO 000040	S3.8BC= 000100	TS.CKP= 000200	T.CPCB 000004	T2.AST= 100000
S.TNRP 000032	S4.ABD= 100000	TS.CKR= 000100	T.CSR 000022	T2.CAF= 000400
S.TTEC 000041	S4.ANI= 000400	TS.EXE= 100000	T.DELT 000056	T2.CHK= 020000
S.WFA 000016	S4.AVO= 001000	TS.HLD= 002000	T.DIN 000044	T2.CKD= 010000
S.XDE 000036	S4.BLK= 002000	TS.MSG= 020000	T.DOUT 000042	T2.DST= 040000
S.XID 000002	S4.DEC= 004000	TS.NRP= 010000	T.DPRI 000040	T2.FXD= 002000
S.XMB 000030	S4.DLO= 000100	TS.OUT= 000400	T.EFLG 000022	T2.HLT= 000200
S.XMP 000024	S4.EDT= 010000	TS.RDN= 040000	T.ERRC 000020	T2.HRT= 001000
S.XMT 000012	S4.EFF= 000020	TS.RUN= 004000	T.IOC 000003	T2.SEF= 004000
S.XMTB 000076	S4.HFL= 000007	TS.STP= 001000	T.ITM 000001	T2.SPN= 000004
S.XMTC 000062	S4.HHT= 000040	TX.NXM= 000002	T.LBF 000046	T2.STP= 000020
S1.DEC= 001000	S4.HSY= 000200	TX.OVR= 000001	T.LBN 000041	T2.WFR= 000001
S1.DPR= 000400	S4.RGS= 020000	TX.TMO= 000004	T.LDV 000044	T3.ACP= 100000
S1.DSI= 004000	S4.SFC= 040000	TY\$CHA= 000400	T.LNK 000000	T3.CAL= 000100
S1.ESC= 000002	S4.VFL= 000010	TY\$EVE= 001000	T.LTM 000054	T3.CLI= 001000
S1.IBF= 002000	S5.ABP= 100000	TY\$STA= 002000	T.MXSZ 000050	T3.GFL= 000010
S1.IBY= 000200	S5.BCC= 020000	TY\$SUM= 000000	T.NAM 000006	T3.MCR= 004000
S1.OBY= 000100	S5.DAD= 040000	TSFLAG 000044	T.NDM 000066	T3.NET= 000020
S1.PTH= 000010	S5.HPC= 000014	TSLIF 000013	T.OFF 000066	T3.NSD= 000200
S1.RES= 010000	S5.HPO= 000020	TSLJFL 000013	T.PCB 000046	T3.PMD= 040000
S1.RNE= 000020	S5.ITI= 000100	TSLIFO 000013	T.PIND 000007	T3.PRV= 010000
S1.RNF= 020000	S5.OXF= 000040	TSLIFS 000013	T.PRI 000002	T3.REM= 020000
S1.RSP= 000004	S5.RPO= 002000	TSLIN 000000	T.QRAC 000060	T3.ROV= 000040
S1.RST= 000001	S5.SW1= 000001	TSLIPS 000006	T.QRIN 000062	T3.RST= 000400
S1.TNE= 040000	S5.TMM= 000002	TSLLD 000012	T.QRPD 000064	T3.SLV= 002000
S1.TSY= 000040	S5.VER= 010000	TSLLDC 000045	T.QUE 000024	T3.SWS= 000302
S1.USI= 100000	S5.XOF= 000004	TSLLDD 000012	T.QVZ = 000026	UC.ALG= 000200
S2.BEL= 020000	S5.XON= 000010	TSLLDO 000012	T.RBF 000050	UC.ATT= 000010
S2.BRO= 000020	S6.EIO= 000400	TSLLDS 000012	T.RBTL 000020	UC.KIL= 000004
S2.CR = 002000	S6.RDI= 100000	TSLLEN 000046	T.RCVL 000012	UC.LGH= 000003
S2.CTO= 040000	S6.RLU= 001000	TSLOPR 000002	T.RDAT 000034	UC.NPR= 000100
S2.CTS= 100000	TEMP1 001034R	TSLTCL 000024	T.RRFL 000072	UC.PWF= 000020
S2.ELF= 001000	TEMP2 001062R	TSLTIM 000026	T.RTM 000055	UC.QUE= 000040
S2.FLF= 000400	TEMP3 001202R	TSLTTP 000014	T.RTY 000030	UD.UNS= 000000
S2.IRO= 000200	TEMP4 001224R	TSLTPS 000020	T.SAST 000054	UD.160= 000004
S2.OBF= 004000	TEMP5 001244R	TSNAPL 000004	T.SEL 000040	UD.200= 000001
S2.ORO= 000100	TEMP6 001554R	TSNFE 000000	T.SELT 000052	UD.556= 000002
S2.PCU= 010000	TEMP7 001556R	TSNLEN 000010	T.SFNC 000014	UD.625= 000005
S2.RCU= 000001	TEMP8 001560R	TSNNUL 000002	T.SITC 000072	UD.8K = 000006
S2.SRO= 000040	TF.FFE= 000004	TSNOPL 000006	T.SRCT 000071	UD.800= 000003
S2.WAL= 000010	TF.ROF= 000001	TSNRNI 000042	T.STAT 000002	UM.CLI= 000036
S2.WRA= 000006	TF.RTO= 000002	TSNRPL 000005	T.ST2 000034	UM.DSB= 000200
S2.WRB= 000002	TF.TAE= 000010	TSNRUL 000007	T.ST3 000036	UM.NBR= 000400
S3.ACR= 000001	TF.TOF= 000004	TSNVR 000001	T.TCBL 000030	UM.OVR= 000001
S3.CTC= 000004	TF.TTO= 000010	TSRPR1 000040	T.TIMC 000016	US.ABO= 000001
S3.FDX= 000200	TF.URF= 000001	TS\$VC 000034	T.TIO 000057	US.BSP= 000002
S3.ICF= 001000	TF.USA= 000002	TS\$5 000030	T.TKSZ 000060	US.BSY= 000200
S3.MHE= 000400	TIATT 001616R	TS\$6 000032	T.TMR 000000	US.FOR= 000040

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 11 F 9
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
NMSG6	012520 R	26-1688 #26-1853
NMSG6A	012636 R	26-1695 #26-1857
NMSG6B	012734 R	26-1700 #26-1860
NMSG7	013014 R	26-1710 #26-1863
NMSG8	013040 R	26-1754 #26-1866
NMSG9	013076 R	26-1766 #26-1869
NMS12A	013314 R	26-1658 #26-1881
NMS2A	012166 R	26-1470 26-1594 #26-1837
NMS5A	012404 R	26-1643 #26-1849
NODSAV	001032 R	#6-244
NOINFO	001022 R	#6-237 31-2368 33-2484 35-2671 37-2780 40-3063
NROUT	000426 R	#6-190 26-1676
NSPEC	000411 R	#6-184 22-1288 26-1520
NTINAM	001024 R	#6-241 39-2863 59-4715
NXTBLK	025006 R	9-400 9-413 10-479 10-526 12-725 26-1487 26-1541 26-1613
NXTCIR	025050 R	26-1791 36-2707 #45-3360 #46-3589
NXTDST	024126 R	27-1941 27-1995
NXTDTE	021126 R	40-2927 #41-3133
NXTGRP	021662 R	35-2636 #36-2705
NXTLIN	025120 R	37-2743 #38-2816
NXTRDT	016522 R	28-2173 28-2190 28-2220 46-3390 #47-3422
NSELEN	000054 R	31-2330 #32-2403
N\$LVC	000036 R	26-1630
N\$TLC	000100 R	53-4259 53-4261
N\$VER	000066 R	26-1763
N\$XLEN	000124 R	26-1762 53-4258
N\$SVCT	= *****	22-1307 25-1407 37-2766 42-3179 44-3279 45-3363 59-4747
QERR1	002510 R	9-425 #9-435
OFF	000077 R	#6-131 14-861 26-1778 35-2644 39-2862 48-3477 53-4137 55-4502 59-4714
OF\$INS	= 000001	#5-99 48-3475
OF\$LOG	= 000100	#5-99
OF\$OFF	= 000002	#5-99
OF\$ON	= 000000	#5-99 48-3472
OF\$SMC	= 000200	#5-99
OF\$RLU	= 000100	48-3494
OKNOWN	000037 R	#6-123 9-393
OMSG1	025430 R	48-3454 #48-3513
OMSG2	025454 R	48-3459 #48-3516
OMSG3	025506 R	48-3482 #48-3519
OMSG4	025552 R	48-3484 #48-3522
OMSG5	025616 R	48-3501 #48-3526
OMSG6	025660 R	48-3504 #48-3529
ON	000064 R	#6-129 14-858 26-1774 35-2647 39-2885 48-3469 50-3708 53-4152 55-4513
OP\$ADD	= 000200	#5-99
OP\$ALL	= 000200	#5-99
OP\$CCS	= 000004	#5-99
OP\$CIE	= 004000	#5-99
OP\$CON	= 000400	#5-99 6-273 12-704 13-826 15-883
OP\$COP	= 000001	#5-99
OP\$COS	= 000100	#5-99

MACRO NAME	REFERENCES															
ERBLK\$	#5-56	6-237	8-353	8-357	8-363	9-435	10-548	10-551	11-641	12-761						
	12-764	12-767	12-770	12-773	12-777	12-780	12-783	12-786	12-789	12-792						
	12-795	26-1831	26-1834	26-1837	26-1840	26-1843	26-1846	26-1849	26-1853	26-1857						
	26-1860	26-1863	26-1866	26-1869	26-1872	26-1875	26-1878	26-1881	26-1887	26-1891						
	26-1894	26-1897	26-1900	26-1903	27-2137	28-2241	29-2281	31-2376	31-2379	31-2382						
	31-2385	33-2493	33-2496	33-2499	33-2502	33-2505	33-2508	33-2511	34-2603	34-2608						
	34-2611	34-2614	34-2617	35-2679	35-2685	35-2688	35-2691	37-2791	37-2794	39-2902						
	39-2907	40-3071	40-3074	40-3077	40-3080	40-3083	40-3086	40-3089	40-3092	40-3095						
	40-3098	40-3101	40-3104	40-3107	40-3110	42-3190	42-3195	48-3513	48-3516	48-3519						
	48-3522	48-3526	48-3529	49-3658	49-3661	49-3664	49-3667	50-3724	50-3727	50-3730						
	51-3825	51-3828	51-3831	51-3834	51-3837	51-3840	51-3843	51-3846	51-3849	51-3852						
	52-3917	52-3920	53-4284	53-4287	53-4290	53-4293	53-4296	53-4299	53-4302	53-4305						
	53-4309	53-4312	54-4363	54-4366	54-4369	54-4372	55-4530	55-4533	55-4536	55-4539						
	55-4542	55-4545	55-4548													
ERMSG\$	#5-56	6-236	8-351	8-352	8-355	8-356	8-362	9-434	10-547	10-550						
	11-640	12-760	12-763	12-766	12-769	12-772	12-776	12-779	12-782	12-785						
	12-788	12-791	12-794	12-797	26-1830	26-1833	26-1836	26-1839	26-1842	26-1845						
	26-1851	26-1852	26-1855	26-1856	26-1859	26-1862	26-1865	26-1868	26-1871	26-1874						
	26-1877	26-1880	26-1884	26-1885	26-1886	26-1889	26-1892	26-1895	26-1898	26-1899						
	26-1902	27-2136	28-2240	29-2280	31-2375	31-2378	31-2381	31-2384	33-2492	33-2495						
	33-2498	33-2501	33-2504	33-2507	33-2510	34-2602	34-2605	34-2608	34-2611	34-2614						
	35-2678	35-2681	35-2684	37-2787	37-2790	37-2793	39-2901	39-2904	39-2907	40-3100						
	40-3076	40-3079	40-3082	40-3085	40-3088	40-3091	40-3094	40-3097	40-3100	40-3103						
	40-3106	40-3109	42-3189	42-3194	48-3512	48-3515	48-3518	48-3521	48-3525	48-3528						
	49-3657	49-3660	49-3663	49-3666	50-3723	50-3726	50-3729	51-3824	51-3827	51-3830						
	51-3833	51-3836	51-3839	51-3842	51-3845	51-3848	51-3851	52-3916	52-3919	53-4283						
	53-4286	53-4289	53-4292	53-4295	53-4298	53-4301	53-4304	53-4307	53-4311	54-4362						
	54-4365	54-4368	54-4371	55-4529	55-4532	55-4535	55-4538	55-4541	55-4544	55-4547						
ERRPT\$	#5-56	8-341	8-342	8-343	9-385	9-425	10-457	10-535	10-538	11-573						
	11-629	12-666	12-680	12-684	12-698	12-750	13-839	14-864	15-930	18-1061						
	18-1071	19-1170	26-1436	26-1456	26-1470	26-1476	26-1556	26-1565	26-1569	26-1594						
	26-1600	26-1643	26-1646	26-1658	26-1688	26-1695	26-1700	26-1710	26-1735	26-1749						
	26-1754	26-1766	26-1769	26-1781	26-1797	26-1810	26-1815	26-1819	26-1821	27-1925						
	27-2131	28-2157	28-2234	29-2260	29-2272	31-2342	31-2358	31-2361	31-2368	31-2370						
	33-2449	33-2456	33-2464	33-2467	33-2475	33-2477	33-2484	33-2486	34-2529	34-2574						
	34-2581	34-2589	34-2597	35-2642	35-2665	35-2671	35-2673	37-2754	37-2773	37-2780						
	37-2782	39-2852	39-2896	40-2936	40-2951	40-2966	40-2969	40-2977	40-2989	40-2991						
	40-3001	40-3005	40-3018	40-3024	40-3046	40-3053	40-3056	40-3063	40-3065	42-3171						
	42-3183	48-3454	48-3459	48-3482	48-3484	48-3501	48-3504	49-3557	49-3568	49-3645						
	49-3648	50-3696	50-3702	50-3714	51-3756	51-3783	51-3788	51-3793	51-3796	51-3801						
	51-3806	51-3811	51-3814	52-3877	52-3908	53-3952	53-3960	53-4093	53-4095	53-4127						
	53-4224	53-4227	53-4271	53-4273	54-4336	54-4352	54-4353	54-4355	55-4398	55-4405						
	55-4454	55-4456	55-4492	55-4517	55-4519											
FLTDF\$	#5-57	5-84														
GETAD	#5-56	12-731	22-1308	25-1408	34-2570	37-2707	39-2894	42-3180	44-3280	45-3364						
	53-4021	53-4025	33-4049	53-4053	53-4080	53-4084	53-4262	59-4749								
GETRV	#5-56	10-461	18-1023	22-1298	25-1397	26-1439	26-1484	26-1524	26-1605	26-1630						
	26-1684	26-1706	26-1713	26-1762	26-1763	27-1949	27-2003	27-2091	31-2329	31-2352						
	31-2354	33-2430	34-2553	35-2635	37-2741	39-2880	40-2926	42-3168	49-3577	49-3578						
	53-4208	53-4258	55-4473	58-4682	59-4733	59-4743	60-4778	60-4785	62-4850	62-4853						
NKRDF\$	#5-63	5-64														
NSFDF\$	#5-57	5-85														

270	002017	ERBLK\$	ERR27
271	002022	ERMSG\$	<Event cannot be filtered>
272	002052	ERBLK\$	ERR28
273	002056	ERMSG\$	<Node name already exists>
274	002106	ERBLK\$	ERR29
275	002112	ERMSG\$	<Object cannot be cleared>
276	002142	ERBLK\$	ERR30
277	002146	ERMSG\$	<Line is not multipoint>
278	002174	ERBLK\$	ERR31
279	002200	ERMSG\$	<Invalid parameter, Tributary>
280	002234	ERBLK\$	ERR32
281	002240	ERMSG\$	<Invalid parameter, Multipoint active>
282	002304	ERBLK\$	ERR33
283	002310	ERMSG\$	<Invalid parameter, Multipoint dead>
284	002352	ERBLK\$	ERR34
285	002356	ERMSG\$	<Line in wrong state>
286	002401	ERBLK\$	ERR35
287	002404	ERMSG\$	<Invalid parameter value, Owner>
288	002442	ERBLK\$	ERR36
289	002446	ERMSG\$	<Operation failed, Owner>
290	002475	ERBLK\$	ERR37
291	002500	ERMSG\$	<Event filter block allocation failure>
292	002545	ERBLK\$	ERR38
293	002550	ERMSG\$	<Invalid parameter value, Sink name>
294	002612	ERBLK\$	ERR39
295	002616	ERMSG\$	<Invalid parameter value, Remote node name>
296	002667	ERBLK\$	ERR40
297	002672	ERMSG\$	<Invalid parameter value, Sink node name>
298	002741	ERBLK\$	ERR41
299	002744	ERMSG\$	<Invalid parameter grouping>
300	002776	ERBLK\$	ERR42
301	003002	ERMSG\$	<Parameter not applicable, hello timer>
302	003047	ERBLK\$	ERR43
303	003052	ERMSG\$	<Parameter not applicable, listen timer>
304	003120	ERBLK\$	ERR44
305	003124	ERMSG\$	<X25ACP not installed>
306	003150	ERBLK\$	ERR45
307	003154	ERMSG\$	<X25ACP must be 'fixed' in RSX11S systems>
308	003224	ERBLK\$	ERR46
309	003230	ERMSG\$	<Adjacency data base allocation failure>
310	003276	ERBLK\$	ERR47
311	003302	ERMSG\$	<Deallocation of unused POOL.. failed>
312	003346	ERBLK\$	ERR48
313	003352	ERMSG\$	<Invalid function for RSX-11M/M-PLUS systems>
314	003425	ERBLK\$	ERR49
315			
316	000000	.PSECT	

VENA - VNP ENABLE/DISABLE LINES MACRO V05.03b Wednesday 11-Sep-85 12:06 Page 18
 FNDFR - FIND A FREE CHANNEL

```

805      .SBTTL  FNDFR - FIND A FREE CHANNEL
806      ;+
807      ***FNDFR-FIND A FREE CHANNEL
808
809      INPUTS:
810          R0 = PDV INDEX
811
812      OUTPUTS:
813          C-BIT = SUCCESS/FAILURE
814          R3 = POINTER TO AVAILABLE CHANNEL IN PDV
815          R4 = CHANNEL NUMBER
816          R1,R2 = DESTROYED
817      ;
818      ;
819      ;
820      FNDFR:
821          002136      010003      MOV      R0,R3      ; COPY PDV INDEX
822          002140      066703      ADD      $PDVTA,R3    ; COMPUTE ADDRESS OF THE PDV
823          002144      011303      MOV      (R3),R3      ; GET THE PDV ADDRESS
824          002146      032763      BIT      #ZF.LLC,Z.FLG(R3) ; IS THIS PROCESS AN LLC ?
825          002154      001416      BEQ      25$          ; IF EQ, YES .. ERROR
826          002156      005002      CLR      R2           ; GET NUMBER OF OF CHANNELS
827          002160      156302      BISB     Z.LLN(R3),R2  ;
828          002164      001412      BEQ      25$          ; IF EQ, NO CHANNELS PERIOD
829          002166      062703      ADD      #Z.MAP,R3     ; POINT TO CHANNELS
830          002172      005004      CLR      R4           ; INITIALIZE CHANNEL COUNT
831
832          002174      012346      10$:      MOV      (R3)+,-(SP) ; GET AN ENTRY
833          002176      006326      ASL      (SP)+          ; IS IT FREE ?
834          002200      103001      BCC      20$          ; IF CC, NO .. KEEP LOOKING
835          002202      100005      BPL      30$          ; IF PL, YES .. GOOD
836          002204      005204      20$:      INC      R4           ; SKIP TO NEXT CHANNEL
837          002206      005302      DEC      R2           ; ANY MORE CHANNELS ?
838          002210      003371      BGT      10$          ; IF GT, YES
839          002212      000261      25$:      SEC          ; INDICATE FAILURE
840          002214      000401      BR       40$          ; AND RETURN
841
842          002216      005743      30$:      TST      -(R3)      ; ACCOUNT FOR PREVIOUS PIP (CLEAR C-BIT)
843          002220      40$:      RETURN

```

VENA - VNP ENABLE/DISABLE LINES MACRO V05.03b Wednesday 11-Sep-85 12:06 Page 19
 FNDFR - FIND A FREE CHANNEL

\$VLIN - SET LOOPBACK NODE BLOCK

```

1366 .SBTTL $VLIN - SET LOOPBACK NODE BLOCK
1367
1368 *** - $VLIN - SET LOOPBACK NODE BLOCK
1369
1370 INPUTS:
1371 RQB+LR.NAM = NODE NAME
1372 RQB+LO.NAM = CHANNEL NUMBER
1373
1374 OUTPUTS:
1375 R0-R2 = DESTROYED
1376
1377
1378 .ENABL LSB
1379 $VLIN::
1380 004234 MOV #RQB+LR.NAM,R1 ; GET ADDRESS OF REMOTE NODE NAME
1381 004234 012701 000012G BIT #CM$SET,$CMAND ; IS THIS A SET COMMAND ?
1382 004246 032767 000000G 000000G BNE 5$ ; IF NE, YES
1383
1384 ; CLEAR REQUEST
1385
1386 004250 CALL $FN$NAM ; TRY TO FIND THE NODE BLOCK
1387 004254 103002 BCC 1$ ; IF SUCCESS - BRANCH
1388 004256 000167 000362 JMP 103$ ; IF NOT - ERROR
1389 004262 016702 000010G 1$: MOV $BFR+R.ADD,R2 ; GET CHANEL NUMBER
1390
1391 004266 005767 000012G TST $BFR+R.FLAG ; CHECK IF LOOPBACK
1392 004272 100402 BMI 3$ ; IF LOOPBACK - BRANCH
1393 004274 000167 000324 JMP 101$ ; IF PL, NOT A LOOPBACK .. ERROR
1394 004300 3$: CALL CLRBIT ; CLEAR LOOPBACK BIT IN SLT
1395 004304 103001 REMNOD ; IF SUCCESS - BRANCH
1396 004306 000545 BCC 30$ ; IF CS, ERROR
1397
1398 004310 REMNOD: SWTCHI #SHMBUF ; USE TEMPORARY BUFFER
1399 004316 GETRV $DECT,#D$END ; READ DECNET HOME BLOCK
1400 004336 SWTCHI ; RESET DEFAULT BUFFER
1401
1402 004344 022701 000004G CMP #SHMBUF+D$RNN+2,R1 ; IS PREVIOUS THE LISTHEAD ?
1403 004350 001022 BNE 2$ ; IF NE, OKAY
1404 004352 016767 000002G 000004G MOV $BFR,$HMBUF+D$RNN+2 ; ELSE, SET NEW NEXT
1405
1406 004360 SWTCHO #SHMBUF ; USE TEMPORARY BUFFER
1407 004366 PUTRC $DECT,#D$END ; WRITE DECNET HOME BLOCK
1408 004406 SWTCHO ; RESET DEFAULT BUFFER
1409
1410 004414 000404 BR 4$ ; AND DEALLOCATE BLOCK
1411
1412 004416 010146 2$: MOV R1,-(SP) ; GET UNMAPPED ADDR OF PREVIOUS BLOCK
1413 004420 010046 MOV R0,-(SP) ; GET UNMAPPED ADDRESS OF CURRENT BLOCK
1414 004422 CALL $XLINK ; UNLINK BLOCK FROM LIST
1415 004426 012701 000014 4$: MOV #R.LEN,R1 ; GET LENGTH OF REMOTE BLOCK
1416 004432 CALL $DEPOL ; DEALLOCATE THE REMOTE BLOCK
1417 004436 000471 BR 30$ ; AND LEAVE
1418
1419 ; SET LOOPBACK REQUEST
1420
1421 004440 5$: CALL $FN$NAM ; TRY TO FIND THE NODE BLOCK
1422 004444 103427 BCS 10$ ; IF CS, NOT FOUND

```

```

1856                                     .SBTTL $TPASW - SET/CLEAR TRANSMIT PASSWORD
1857                                     :+
1858                                     *** - $TPASW - SET/CLEAR TRANSMIT PASSWORD
1859                                     :
1860                                     INPUTS:
1861                                     $CMAND = SET/CLEAR
1862                                     :
1863                                     OUTPUTS:
1864                                     APPROPRIATE ACTION TAKEN
1865                                     :-
1866
1867 006260                                     $TPASW::
1868 006260                                     CALL      CHKXPT          ; BE SURE XPT IS LOADED
1869 006264 103450                             BCS      40$          ; IF CS, XPT NOT LOADED
1870 006266                                     GETRV     R2,#N$XLEN      ; READ IN POINTERS
1871 006304 016702 000070G                     MOV     N$VER+2+$BFR,R2 ; ADDRESS OF TRNSMIT/RECEIVE PASSWORDS
1872 006310                                     GETRV     R2,#18.        ; FLAG AND PASSWORDS
1873
1874 006326 032767 000000G 000000G             BIT      #CM$SET,$CMAND ; SET PASSWORD ?
1875 006334 001415                                     BEQ      20$          ; NO
1876
1877 006336 012704 000012G 000000G             MOV     #V$XMT+$BFR,R4 ; SET DESTINATION ADDRESS
1878 006342 052767 040000 000000G             BIS     #V$XMT,V$FLG+$BFR ; INDICATE VALID TRANSMIT PASSWORD
1879 006350 012705 000020G                     MOV     #RQB+LO.TPA,R5 ; POINT AT PASSWORD SOURCE
1880 006354 012703 000010                     MOV     #8,R3          ; NUMBER OF CHARACTERS TO MOVE
1881 006360 112524                                     MOVB     (R5)+,(R4)+ ; MOVE STRING
1882 006362 005303 10$:                         DEC     R3
1883 006364 001375                                     BNE     10$          ; TILL DONE
1884 006366 000405                                     BR       30$          ; FINISH UP
1885
1886                                     :
1887                                     : CLEAR PASSWORD
1888 006370 042767 040000 000000G 20$:          BIC     #V$XMT,V$FLG+$BFR ; CLEAR FLAG BIT
1889 006376 005067 000012G                     CLR      V$XMT+$BFR ; FORCE ILLEGAL PASSWORD
1890
1891 006402 30$:                                PUTRC
1892 006406 40$:                                RETURN ; REWRITE DATA BLOCK

```

```

2477          .SBTTL $VLOG - PROCESS LOGGING REQUEST
2478
2479      ;+
2480      **-$VLOG-PROCESS LOGGING REQUEST
2481
2482      INPUTS:
2483          $OPTION = OPTIONS WORD
2484
2485      OUTPUTS:
2486          C-BIT = SUCCESS/FAILURE
2487      ;+
2488
2489      $VLOG::
2490          012002      MOV     #*PREVL,R0          ; GET EVL'S PROCESS NAME
2491          012002      CALL    $FPDV              ; IS EVL IN SYSTEM?
2492          012012      BCS     101$               ; BR IF NO - EVENT LOGGING NOT SUPPORTED
2493
2494          012014      032767      000000G 000000G      BIT     #CM$SET,$CMAND          ; CLE COMMAND ?
2495          012022      001013      BNE     5$          ; IF NO - BRANCH
2496          012034      032767      001000      000000G      BIT     #OP$KEV,$OPTION        ; KNOWN EVENTS
2497          012032      001407      BEQ     5$          ; IF NO - BRANCH
2498          012034      032767      000040      000000G      BIT     #OP$LNE,$OPTION        ; LINE QUALIFIER ?
2499          012042      001403      BEQ     5$          ; IF NO - BRANCH
2500          012044      CALL     LINFLT            ; CLEAR EVENTS SPECIFIED TO THIS LINE
2501
2502      RETURN
2503
2504      5$:
2505      ; CHECK FOR VALID LOGGING PARAMETERS
2506
2507          012052      012705      000240'      MOV     #PVALID,R5          ; POINT TO PARAMETER VALIDITY TABLE
2508          012056      012501      10$:      MOV     (R5)+,R1          ; GET BIT TO CHECK FOR
2509          012060      001410      BEQ     30$          ; BR IF END OF TABLE
2510          012062      030167      000000G      BIT     R1,$OPTION        ; IS THIS PARAMETER PRESENT?
2511          012066      001403      BEQ     20$          ; BR IF NO
2512          012070      CALL     @ (R5)            ; CHECK FOR PARAMETER VALIDITY
2513          012074      103416      BCS     60$          ; BR IF ERROR
2514
2515          012076      005725      20$:      TST     (R5)+          ; SKIP OVER ADDRESS OF ROUTINE
2516          012100      000766      BR     10$          ; CHECK NEXT PARAMETER
2517
2518      ; OPERATE LOGGING PARAMETERS
2519
2520          012102      012705      000262'      30$:      MOV     #CHTBLE,R5          ; POINT TO DISPATCH TABLE
2521          012106      012501      40$:      MOV     (R5)+,R1          ; GET BIT TO CHECK FOR
2522          012110      001410      BEQ     60$          ; BR IF END OF TABLE
2523          012112      030167      000000G      BIT     R1,$OPTION        ; OPERATE ON THIS PARAMETER?
2524          012116      001403      BEQ     50$          ; BR IF NO
2525          012120      CALL     @ (R5)            ; CALL OPERATE ROUTINE
2526          012124      103402      BCS     60$          ; BR IF FAILURE
2527
2528          012126      005725      50$:      TST     (R5)+          ; POINT PAST ROUTINE
2529          012130      000766      BR     40$          ; AND CHECK FOR NEXT PARAMETER
2530          012132      60$:      RETURN
2531
2532      ; ERROR CONDITIONS
2533

```

```

3003 .SBTTL CKSINK - CHECK VALIDITY OF SINK NODE ID
3004
3005 ;+
3006 CKSINK - CHECK VALIDITY OF SINK NODE ID
3007
3008 INPUTS:
3009 LO,SKN+RQB - SINK NODE ID
3010 $VFLAG - FLAGS WORD
3011
3012 OUTPUTS:
3013 SINKND - SINK NODE ADDRESS IF VALID
3014 CARRY - SUCCESS/FAILURE
3015
3016
3017
3018
3019 CKSINK: SWITCHI #SHMBUF ; SET UP INPUT BUFFER
3020 GETRV $DECT,$DSEND ; READ DECNET HOME BLOCK
3021 MOV $HMBUF+$D$NUM,-(SP) ; STORE NODE NUMBER
3022 SWITCHI
3023
3024 MOV #RQB+LO,SKN,R1 ; POINT TO SINK NODE ID
3025 BIT #VF.SKA,$VFLAG ; NODE ADDRESS SPECIFIED?
3026 BEQ 10$ ; BR IF NO
3027 MOV (R1),R4 ; SAVE SINK NODE ADDRESS
3028 BR 20$ ; CONTINUE
3029 10$: CALL $FNAM ; IS THERE A REMOTE NODE NAME BLOCK?
3030 BCC 15$ ; IF FOUND - BRANCH
3031 TST (SP)+ ; CLEAN UP STACK
3032 BR 101$ ; BR IF NO
3033 15$: MOV $BFR+R.ADD,R4 ; SAVE SINK NODE ADDRESS
3034 20$: CMP (SP)+,R4 ; IS THIS THE EXECUTOR NODE?
3035 BEQ 30$ ; BR IF YES
3036 MOV R4,SINKND ; STORE SINK NODE ADDRESS
3037 CLC ; INDICATE SUCCESS
3038 40$: RETURN
3039
3040 ; ERROR MESSAGES
3041
3042 101$: ERRPT$ ERR41,$EROUT

```

```

1329 .SBTTL CKPREV
1330
1331 ;+
1332 ; CKPREV - CHECK TO SEE IF PREVIOUS EVENT WAS SET AND IF IT WAS, DISPLAY IT
1333 ;
1334 ; INPUTS:
1335 ;   FLAGS - FLAGS BYTE
1336 ;   PREVNT - VALUE OF PREVIOUS EVENT TYPE IF PREVIOUS EVENT TYPE WAS SET
1337 ;
1338 ; OUTPUTS:
1339 ;   FLAGS - F.PREV IS CLEARED
1340 ;   THE PREVIOUS EVENT VALUE IS DISPLAYED IF NECESSARY
1341 ;
1342 ;-
1343
1344 007174 132767 000002 172372 CKPREV: BITB #F.PREV,FLAGS ; WAS PREVIOUS EVENT TYPE SET
1345 007202 001411          BEQ 20$ ; BR IF NO
1346 007204 116701 171775      MOVB PREVNT,R1 ; GET EVENT TYPE NUMBER TO DISPLAY
1347 007210 001403          BEQ 10$ ; BR IF FIRST NUMBER IN RANGE
1348 007212 005002          CLR R2 ; SUPPRESS ZEROES
1349 007214          CALL $CBDMG ; CONVERT TO ASCII
1350 007226 142767 000002 172346 10$: BICB #F.PREV,FLAGS ; INDICATE THIS EVENT TYPE NOT SET
1351 007226          20$: RETURN

```

[illegible]

```

2513 .SBTTL DMXPR - SHOW MODULE X25-PROTOCOL
2514
2515 ;+
2516 ; DMXPR - SHOW MODULE X25-PROTOCOL
2517
2518 ; INPUTS:
2519 ; RQB - ADDRESS OF REQUEST BLOCK
2520
2521 ; OUTPUTS:
2522 ; INFORMATION DISPLAYED ON USER'S TERMINAL
2523
2524 ;-
2525
2526 DMXPR: MOV #*RPLI,RO ; GET PLI'S PROCESS NAME
2527 CALL FNDPDV ; LOOK FOR PDV
2528 BCC 10$ ; BR IF FOUND IT
2529 ERRPT$ XPER1,$EROUT ; DISPLAY ERROR MESSAGE
2530
2531 10$: MOV RO,PLIPDV ; SAVE PLI'S PDV ADDRESS
2532 BIT #OP$DTE!OP$KDT,$OPTHON ; WAS DTE QUALIFIER SPECIFIED?
2533 BEQ 20$ ; BR IF NO
2534 CALL DMDTE ; DISPLAY DTE INFO
2535 JMP 100$ ; AND EXIT ;[TD001]
2536 ;**~1
2537 20$: BIT #OP$GRP!OP$KGR,$OPTHON ; WAS GROUP QUALIFIER SPECIFIED?
2538 BEQ 30$ ; BR IF NO
2539 CALL DMGRP ; DISPLAY GROUP INFO
2540 JMP 100$ ; AND EXIT ;[TD001]
2541 ;**~1
2542 ; DISPLAY X25-PROTOCOL, NO QUALIFIERS
2543
2544 30$: MOV RO,R5 ; Save PDV address ;[TD001]
2545 MOV $PSIPT,RO ; Get PSI home block address ;[TD001]
2546 BIT #PF.XDF,$PFLAG ; Check for extended data format ;[TD001]
2547 BEQ 40$ ; Use old-format parameters if unset ;[TD001]
2548 ADD #H$GNAM,RO ; Address extended network name ;[TD001]
2549 MOV #MAXID,R2 ; Max length of extended name ;[TD001]
2550 BR 50$ ;[TD001]
2551 40$: ADD #H$NETW,RO ; Address old-format network name ;[TD001]
2552 MOV #6,R2 ; Length of old-format network name ;[TD001]
2553 GETRV RO,R2 ; Read in network name ;[TD001]
2554 MOV #TEMP1,RO ; Point temporary storage ;[TD001]
2555 MOV #SBFR,R1 ; Point to network name ;[TD001]
2556 MOV (R1)+,(R2)+ ; Store network name ;[TD001]
2557 SOB R2,60$ ; ... ;[TD001]
2558 60$: CMP RO,#TEMP1 ; Remove trailing spaces ;[TD001]
2559 BLOS 80$ ;[TD001]
2560 -(RO),#SPACE ;[TD001]
2561 BEQ 70$ ;[TD001]
2562 INC RO ;[TD001]
2563 70$: CLRB (RO)+ ; Create asciz string ;[TD001]
2564 ;**~10
2565 MOV Z,DAT(R5),RO ; GET LINE TABLE ADDRESS
2566 CLR $RBLCK ; ASSUME LINE TABLE IS IN DSR
2567 CMP #120000,RO ; IS LINE TABLE IN PROCESS SPACE?
2568 BHI 90$ ; BR IF NO
2569 MOV Z,DSP(R5),$RBLCK ; GET PROCESS MAPPING BIAS

```

```

3080 023522          ERBLK$ DSMS2C,<TEMP1,TEMP2>          ;[TD001]
3081                                     ;[TD001]
3082 023532          ERMSG$ < Call mask = %A, Call value = %A>
3083 023575          ERBLK$ DSMS3,<TEMP1,TEMP2>
3084
3085 023604          ERMSG$ < DTE = %A>
3086 023620          ERBLK$ DSMS4,<TEMP1>
3087
3088 023626          ERMSG$ < Group name = %A>          ; Bug fix          ;[TD001]
3089 023651          ERBLK$ DSMS5,<TEMP1>          ;***-1
3090
3091 023656          ERMSG$ < Subaddress = %F%N>
3092 023703          ERBLK$ DSMS6,<TEMP1>
3093
3094 023710          ERMSG$ < Subaddress = %F-%F%N>
3095 023740          ERBLK$ DSMS7,<TEMP1,TEMP2>
3096
3097 023750          ERMSG$ < Destination not in system%N>
3098 024004          ERBLK$ DSMS8
3099
3100 024010          ERMSG$ < Node = %A>          ; Extended fields          ;[TD001]
3101 024025          ERBLK$ DSMS9,<TEMP1>          ;[TD001]
3102                                     ;[TD001]
3103 024032          ERMSG$ < User = %A>          ;[TD001]
3104 024047          ERBLK$ DSMS10,<TEMP1>          ;[TD001]
3105                                     ;[TD001]
3106 024054          ERMSG$ < Password set>          ;[TD001]
3107 024074          ERBLK$ DSMS11          ;[TD001]
3108                                     ;[TD001]
3109 024100          ERMSG$ < Account = %A>          ;[TD001]
3110 024120          ERBLK$ DSMS12,<TEMP1>          ;[TD001]

```


VDIS - VNP SHOW COMMANDS
DISALI - DISPLAY ALIAS INFO

MACRO V05.03b Monday 15-Jul-85 12:14^{6 5} Page 49-2

```
3647
3648 026264      110$:  ERRPT$  AMSG4,MSGOUT      ; DISPLAY TERMINAL SCOPE AND DESTINATION
3649
3650 026274      120$:  RETURN
3651
3652              .ENABL  LC
3653
3654      :
3655      : ERROR MESSAGES
3656      :
3657 026276      ERMSG$ <%N%A %A as of %Y%N>
3658 026320      ERBLK$ AMSG1,<TEMP2,TEMP1>
3659
3660 026330      ERMSG$ <Alias name = %A%N>
3661 026351      ERBLK$ AMSG2,<TEMP1>
3662
3663 026356      ERMSG$ < Scope = Global, Destination = %A%N>
3664 026423      ERBLK$ AMSG3,<TEMP2>
3665
3666 026430      ERMSG$ < Scope = Terminal %A%B:, Destination = %A%N>
3667 026505      ERBLK$ AMSG4,<TEMP3,TEMP1,TEMP2>
3668
3669              .DSABL  LC
```

4209	031560	012703	000004G	MOV	#\$BFR+\$RNN+2,R3	; STORE REMOTE LISTHEAD
4210						
4211	031564	012705	000014	MOV	#R.LEN,R5	; GET SIZE OF REMOTE NODE BLOCK
4212	031570			CALL	NY\$BLK	; GET NEXT NODE NAME BLOCK
4213	031574	103425		BCS	470\$; BR IF NOT THERE - DISPLAY STATE
4214	031576	026304	000010	CMP	R,ADD(R3),R4	; IS THIS THE ONE?
4215	031602	001372		BNE	450\$; BR IF NO
4216	031604	012704	001062	MOV	#TEMP2,R4	; ELSE, DISPLAY NAME AND STATE
4217	031610	062703	000002	ADD	#R.NAM,R3	; POINT TO NODE NAME
4218		000003		.REPT	3	
4219				MOV	(R3)+,(R4)+	; STORE THE NAME
4220				.ENDR		
4221	031622	122744	000040	460\$: CMPB	#SPACE,-(R4)	; DELETE TRAILING BLANKS
4222	031626	001775		BEQ	460\$	
4223	031630	112764	000000 000001	MOVB	#0,1(R4)	; CREATE ASCIZ STRING
4224	031636			ERRPT\$	MSG5,MSGOUT	; DISPLAY LOOPBACK NODE NAME & STATE
4225	031646	000404		BR	480\$; FINISHED
4226						
4227	031650			470\$: ERRPT\$	MSG6,MSGOUT	; DISPLAY STATE
4228						
4229						
4230						
4231	031660	016600	000012	480\$: MOV	12(SP),R0	; RESTORE R0
4232	031664	016602	000006	MOV	6(SP),R2	; RESTORE R2
4233	031670	032760	040000 000000	BIT	#LF.RDY,L.FLG(R0)	; IS LINE LOADED?
4234	031676	001507		BEQ	540\$; BR IF NO - DONE
4235	031700	116001	000021	MOVB	L.OWNR(R0),R1	; ASSUME OWNER = LINE OWNER
4236	031704	105760	000014	TSTB	L.NSTA(R0)	; MULTIPOINT LINE?
4237	031710	001411		BEQ	490\$; IF EQ, NO
4238	031712	010001		MOV	R0,R1	; GET SLT ADDRESS
4239	031714	062701	000022	ADD	#L.MPF,R1	; POINT TO STATION TABLE
4240	031720	010246		MOV	R2,-(SP)	; GET TRIB NUMBER
4241						
4242	031722	006316		ASL	(SP)	; STATION TABLES ARE 4 BYTES
4243	031724	006316		ASL	(SP)	
4244	031726	062601		ADD	(SP)+,R1	; ...
4245	031730	116101	000003	MOVB	S.OWNR(R1),R1	; ELSE, GET TRIB OWNER
4246	031734	066701	000000G	ADD	\$PDVTA,R1	; POINT TO PDV TABLE ENTRY
4247	031740	011101		MOV	(R1),R1	; GET THE PDV ADDRESS
4248	031742	022761	114224 000000G	CMP	#*RXPT,Z.NAM(R1)	; IS THE OWNER XPT?
4249	031750	001062		BNE	540\$; IF NE, NOTHING TO DISPLAY
4250	031752	005046		CLR	-(SP)	; CLEAR STACK FOR SLN
4251	031754	016703	000000G	MOV	\$SLTMA,R3	; GET ADDRESS OF FIRST SLT
4252	031760	022300		500\$: CMP	(R3)+,R0	; IS THIS THE CORRECT ENTRY?
4253	031762	001402		BEQ	510\$; BR IF YES
4254	031764	005216		INC	(SP)	; ELSE UPDATE SLN
4255	031766	000774		BR	500\$; AND CONTINUE
4256	031770	110266	000001	510\$: MOVB	R2,1(SP)	; SET TRIB NUMBER ONTO HIGH BYTE
4257	031774	012605		MOV	(SP)+,R5	; RETRIEVE SLN/STA
4258	031776			GETRV	Z.DAT(R1),#N\$XLEN	; GET XPT'S DATA BASE
4259	032016	016767	000102G 000000G	MOV	N\$TLC+2+\$BFR,\$RBLCK	; MAP TO CIRCUIT COUNTRIES
4260	032024	016700	000100G	MOV	N\$TLC+\$BFR,R0	; GET THE NUMBER OF LINE COUNTERS
4261	032030	016701	000104G	MOV	N\$TLC+4+\$BFR,R1	; POINT TO THE FIRST
4262	032034			520\$: GETAD	R1,#T\$LEN	; READ IN COUNTER BLOCK
4263	032052	005300		DEC	R0	; AT END OF COUNTERS?
4264	032054	002420		BLT	540\$; IF LT, NOTHING TO DISPLAY
4265	032056	026705	000000G	CMP	T\$LIN+\$BFR,R5	; ARE THESE THE COUNTERS?

4755 035036 005302
 4756 035040 003375
 4757 035042 112711 000000
 4758 035046
 4759 035050

DEC R2
 BGT 20\$
 MOVB #0,(R1)
 RESG <R5>
 RETURN

; DECREMENT COUNT
 ; BR IF MORE TO STORE
 ; CREATE AN ASCIZ STRING
 ; RESTORE R5

US.FRK= 000002
US.KPF= 000001
US.LAB= 000004
US.MDE= 000002
US.MDM= 000020
US.MNT= 000100
US.OFL= 000001
US.PUB= 000004
US.PWF= 000010
US.RED= 000002
US.SHR= 000001
US.SPU= 000002
US.UMD= 000010
US.VV = 000001
US.WCK= 000010
UU.ABO= 000400
UU.ATN= 000100
UU.AVN= 000004
UU.GUS= 000010
UU.IOS= 002000
UU.ONL= 000020
UU.RCT= 000002
UU.RDY= 000200
UU.SER= 000001
UU.SIO= 001000
UU.SPC= 000040
U.ACB = 000062
U.ACP = 000032
U.ADMA = 000066
U.AFLG = 000064
U.ATT = 000022
U.BPKT= 000044
U.BUF = 000024
U.CBF = 000032
U.CLI = 177772
U.CNT = 000030
U.COTQ = 000030
U.CTCB = 000026
U.CTL = 000004
U.CTYP = 000052
U.CW1 = 000010
U.CW2 = 000012
U.CW3 = 000014
U.CW4 = 000016
U.CYL = 000104
U.DCB = 000000
U.FCDE= 000042
U.FNUM= 000040
U.GRP = 000102
U.KCSR= 000032
U.KCS6= 000034
U.LEN = 000042
U.LUIC = 177774
U.MEDI= 000040
U.MLUN= 000050
U.MUP = 177772
U.OWN = 177776
U.RBNS= 000112
U.RCTC= 000113
U.RCTS= 000110
U.RED = 000002
U.RED2 = 000034
U.SCB = 000020
U.SHST= 000076
U.SHUN= 000074
U.SPC = 000036
U.STS = 000005
U.ST2 = 000007
U.SUB = 000036
U.TCHP = 000042
U.TCVP = 000043
U.TFLK = 000040
U.TFRQ = 000036
U.TIXL = 000060
U.TLPP = 000053
U.TMT1 = 000047
U.TRCK= 000100
U.TSTA = 000026
U.TST5 = 000054
U.TST6 = 000056
J.TTAB = 000050
U.TTYP = 000046
U.TUX = 000024
U.UHVR= 000107
U.UIC = 000044
U.UNFL= 000052
U.UNIT = 000006
U.UNSZ= 000114
U.UNTI= 000060
J.USVR= 000106
U.UTIL= 000036
U.VCB = 000034
U.VSER= 000120
U.2MED= 000070
U2.AT = 000020
U2.CRT= 002000
U2.DH1= 100000
U2.DJ1= 040000
U2.DZ1= 000100
U2.ESC= 001000
U2.HFF= 010000

U2.HLD= 000040
U2.LOG= 000400
U2.LWC= 000001
U2.L3S= 000004
U2.L8S= 010000
U2.NEC= 004000
U2.PRIV= 000010
U2.RMT= 020000
U2.R04= 100000
U2.SCS= 000004
U2.SLV= 000200
U2.VT5= 000002
U2.7CH= 010000
U3.OPA= 100000
U3.PAR= 040000
U3.UPC= 020000
U4.CR = 000100
VF.RCV= 100000
VF.XMT= 040000
VF.ADD= ***** GX
VF.SKA= ***** GX
VF.SS = ***** GX
V\$FLG = 000000
V\$LEN = 000022
V\$RCV = 000002
V\$XMT = 000012
V\$CTR= 001000
WC.CNT= 000400
WC.EVE= 004000
WC.TRI= 002000
WC.UNT= 001000
XPER1 = 020062R
XPMS1 = 020162R
XPMS2 = 020272R
XPMS3 = 020402R
XPMS4 = 020510R
X\$SDBT= 000000
X\$ER1 = 022264R
X\$MS1 = 022334R
X\$ER1 = 024442R
X\$MS1 = 024522R
YF\$ADR= 000100
YF\$DI= 000020
YF\$DIR= 000040
YF\$LOO= 000001
YF\$RES= 000200
YF\$RLR= 000010
YF\$RLU= 000004
YF\$128= 000002
Y\$DPS2 = 000002

Y\$DWSZ = 000006
Y\$FLG = 000000
Y\$LEN = 000020
Y\$LMPK = 000017
Y\$MCLR = 000016
Y\$MPSZ = 000004
Y\$MRES = 000015
Y\$MRST = 000014
Y\$MWSZ = 000007
Y\$TCAL = 000011
Y\$TCLR = 000013
Y\$TRES = 000012
Y\$TRST = 000010
ZF.MUX= ***** GX
Z\$ACPD = 000006
Z\$ACTC = 000016
Z\$CLEN= 000006
Z\$CTIM = 000014
Z\$ICRE = 000024
Z\$LEN = 000026
Z\$LLRE = 000025
Z\$MXAC = 000022
Z\$TCLZ = 000020
Z\$UCB = 000012
Z\$WBAT = 000000
Z\$WEND = 000004
Z.DAT = ***** GX
Z.DSP = ***** GX
Z.FLG = ***** GX
Z.LLN = ***** GX
Z.MAP = ***** GX
Z.NAM = ***** GX
Z.PCB = ***** GX
\$BFR = ***** GX
\$CBDMG= ***** GX
\$CBOMG= ***** GX
\$CBTA = ***** GX
\$CCBNM= ***** GX
\$CEACC= ***** GX
\$CLBUF= ***** GX
\$CSTA = ***** GX
\$DALI = 002512RG
\$DCIR = 014042RG
\$DECP= ***** GX
\$DEVHD= ***** GX
\$DIV = ***** GX
\$DLIN = 015160RG
\$DLOG = 003446RG
\$DMOD = 015540RG
\$DNOD = 007426RG

\$DOBJ = 002272RG
\$DPRO = 003164RG
\$DSYS = 001706RG
\$ERMSG= ***** GX
\$EROUT= ***** GX
\$FILHD= ***** GX
\$FNAM= ***** GX
\$GETAD= ***** GX
\$GETRV= ***** GX
\$HMBUF= ***** GX
\$LGCON= ***** GX
\$LGFNB= ***** GX
\$LGMON= ***** GX
\$LGSTT= ***** GX
\$LGUIC= ***** GX
\$LLCTA= ***** GX
\$MUL = ***** GX
\$OBJJHD= ***** GX
\$OPTON= ***** GX
\$PDVNM= ***** GX
\$PDVTA= ***** GX
\$PFLAG= ***** GX
\$PSIPT= ***** GX
\$QUAL = ***** GX
\$RADDR= ***** GX
\$RBLCK= ***** GX
\$RDBNM= ***** GX
\$RDBSZ= ***** GX
\$RDBTH= ***** GX
\$RSIZE= ***** GX
\$SAVAL= ***** GX
\$SAVRG= ***** GX
\$SDBNM= ***** GX
\$SLTMA= ***** GX
\$SLTNM= ***** GX
\$STIOUT= ***** GX
\$TYPE = ***** GX
\$VFLAG= ***** GX
\$\$ = 177777
\$\$LCNT= 000154 G
\$\$LLIN= 000076 G
\$\$\$ = 034162R
\$\$\$ARG= 000000
\$\$\$OST= 000014
\$\$\$R = 000010
\$\$\$S = 034164R
\$.MGMGE= ***** GX
\$.TLGTH= ***** GX
\$.TSKHD= ***** GX
..OFF.= 177732

. ABS. 177776 000 (RW,I,GBL,ABS,OVR)
035630 001 (RW,I,LCL,REL,CON)
Errors detected: 0

*** Assembler statistics

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 12 G 9
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
OP\$DST	= 000200	#5-99 31-2366 32-2404 39-2854 40-3061 42-3173
OP\$DTE	= 000040	#5-99 34-2532 35-2669
OP\$DUP	= 000002	#5-99
OP\$EST	= 000100	#5-99
OP\$EVE	= 010000	#5-99
OP\$FIL	= 000200	#5-99 6-274 12-704 15-885
OP\$GRP	= 000100	#5-99 34-2537 37-2778
OP\$HOS	= 000001	#5-99
OP\$HTM	= 000002	#5-99
OP\$INA	= 004000	#5-99
OP\$INC	= 001000	#5-99
OP\$KDS	= 002000	#5-99 39-2854 41-3135 42-3173
OP\$KDT	= 000400	#5-99 34-2532 36-2709
OP\$KEV	= 001000	#5-99
OP\$KGR	= 001000	#5-99 34-2537 38-2818
OP\$KNT	= 010000	#5-99 30-2305 33-2433
OP\$KSK	= 002000	#5-99 12-669 12-671 17-977
OP\$LCT	= 000200	#5-99
OP\$LNE	= 000040	#5-99
OP\$LOC	= 000001	#5-99 #5-99
OP\$LST	= 000002	#5-99
OP\$LTM	= 000001	#5-99
OP\$LVL	= 000010	#5-99
OP\$MAC	= 000010	#5-99
OP\$MCO	= 000002	#5-99
OP\$MDE	= 000400	#5-99
OP\$MLI	= 000004	#5-99
OP\$MON	= 000100	#5-99 6-275 12-704 13-822
OP\$MST	= 000020	#5-99
OP\$NAM	= 000004	#5-99 #5-99
OP\$NET	= 004000	#5-99 30-2307 33-2482
OP\$NLI	= 000002	#5-99
OP\$NOD	= 000020	#5-99
OP\$OUT	= 002000	#5-99
OP\$OWN	= 000020	#5-99
OP\$PAR	= 000010	#5-99
OP\$PRI	= 000010	#5-99
OP\$RET	= 010000	#5-99
OP\$RPA	= 000010	#5-99
OP\$SCO	= 000001	#5-99
OP\$SEG	= 000400	#5-99
OP\$SER	= 000400	#5-99
OP\$SIN	= 000010	#5-99 12-667
OP\$STA	= 000004	#5-99
OP\$TPA	= 000020	#5-99
OP\$TRI	= 000040	#5-99
OP\$UCS	= 000020	#5-99
OP\$USE	= 000002	#5-99
OP\$VEC	= 000040	#5-99
OP\$VER	= 000040	#5-99
OP\$XAC	= 000001	#5-99 29-2286
OP\$XPR	= 000002	#5-99 29-2287

MACRO NAME	REFERENCES									
NWDF\$	#5-59	5-86								
OBJDF\$	#5-57	5-87								
OFF\$	#6-267	#6-268								
PCLDF\$	#5-58	5-88								
PHBDF\$	#5-59	5-89								
PLBDF\$	#5-60	5-90								
PLIDF\$	#5-59	5-91								
PVCD\$	#5-60	5-92								
QDPB\$	#6-267	6-267	#6-268	6-268						
QIOW\$	#5-58	6-267	6-268							
RAND	#10-461	10-461	#12-731	12-731	#18-1023	18-1023	#22-1298	22-1298	#22-1308	22-1308
	#25-1397	25-1397	#25-1408	25-1408	#26-1439	26-1439	#26-1484	26-1484	#26-1524	26-1524
	#26-1605	26-1605	#26-1630	26-1630	#26-1684	26-1684	#26-1706	26-1706	#26-1713	26-1713
	#26-1762	26-1762	#26-1763	26-1763	#27-1949	27-1949	#27-2003	27-2003	#27-2091	27-2091
	#31-2329	31-2329	#31-2352	31-2352	#31-2354	31-2354	#33-2430	33-2430	#34-2553	34-2553
	#34-2570	34-2570	#35-2635	35-2635	#37-2741	37-2741	#37-2767	37-2767	#39-2880	39-2880
	#39-2894	39-2894	#40-2926	40-2926	#42-3168	42-3168	#42-3180	42-3180	#44-3280	44-3280
	#45-3364	45-3364	#49-3577	49-3577	#49-3578	49-3578	#53-4021	53-4021	#53-4025	53-4025
	#53-4049	53-4049	#53-4053	53-4053	#53-4080	53-4080	#53-4084	53-4084	#53-4208	53-4208
	#53-4258	53-4258	#53-4262	53-4262	#55-4473	55-4473	#58-4682	58-4682	#59-4733	59-4733
	#59-4743	59-4743	#59-4749	59-4749	#60-4778	60-4778	#60-4785	60-4785	#62-4850	62-4850
	#62-4853	62-4853								
RDTDF\$	#5-60	5-93								
RESRG	#5-56	27-1983	27-2055	27-2119	29-2273	39-2888	40-3015	46-3400	50-3715	51-3816
	53-4274	55-4520	56-4600	57-4664	59-4	63				
RETURN	8-345	9-428	10-540	11-633	12-7	13-41	14-865	15-931	16-956	17-1000
	18-1073	19-1171	20-1201	21-1270	22-1	23-1351	24-1379	25-1414	26-1824	27-2128
	28-2231	29-2276	30-2311	30-2314	31-237	32-2411	33-2487	34-2598	35-2674	36-2723
	37-2783	38-2834	39-2897	40-3066	41-3151	42-3185	43-3233	44-3310	44-3313	44-3333
	44-3335	45-3367	46-3401	47-3429	48-3506	49-367	50-3716	51-3817	52-3909	53-4275
	54-4356	55-4521	56-4601	57-4665	58-4694	59-4	60-4790	61-4829	62-4856	63-4912
	64-4949	65-4984	66-5004							
RNBDF\$	#5-57	5-94								
RNWDF\$	#5-72	5-95								
SAVRG	#5-56	27-1969	27-2020	27-2107	29-2271	39-2860	40-3012	46-3389	50-3693	51-3752
	53-3949	55-4395	56-4569	57-4621	59-4712	63-4871				
SLTDF\$	#5-57	5-96								
SOB	31-2338	33-2447	33-2454	33-2462	33-2473	34-2557	36-2720	37-2747	40-2934	40-2949
	40-2964	40-2975	40-2987	40-3044	54-4350					
SWTCHI	#5-56	10-460	10-462	18-1022	18-1024	22-1297	22-1301	22-1323	25-1396	25-1400
	25-1412	26-1761	31-2350	31-2357	49-3575	49-3588				
TCBDF\$	#5-58	5-97								
TMPDF\$	#5-88	5-88								
UCBDF\$	#5-58	5-98								
VNPDBG	#5-73	27-1922	29-2257							
VNPDF\$	#5-56	5-99								
XPDD\$	#5-60	5-100								
X29DF\$	#5-60	5-101								
.ADDRB	#5-88									
.ADDRW	#5-88									
.APR	#5-88									
.BIN	#5-88									

```

318
319
320
321
322
323
324
325 000000 000000
326 000002 176000 177777 177777
327 000010 177777
328 000012 000002
329 000014 177774 177777 177777
330 000022 177777
331 000024 000003
332 000026 177770 177777 177777
333 000034 177777
334 000036 000004
335 000040 000000 177770 177777
336 000046 177777
337 000050 000005
338 000052 000000 177776 177777
339 000060 177777
340 000062 000006
341 000064 177770 177777 177777
342 000072 177777
343 000074 000100
344 000076 177771 177777 177777
345 000104 177777
346 000106 000104
347 000110 177777 177777 177777
348 000116 177777
349 000120 000105
350 000122 177776 177760 177777
351 000130 177777
352 000132 000135
353 000134 177776 177777 177777
354 000142 177777
355 000144 000136
356 000146 177776 177777 177777
357 000154 177777
358 000156 000340
359 000160 177760 177777 177777
360 000166 177777
361 000170 000341
362 000172 176376 177777 177777
  
```

+
 ***-EVENT CLASS/TYPE VALIDITY TABLE
 THE TABLE IS ORGANIZED AS FOLLOWS:
 .WORD CLASS ; CLASS CODE TO MATCH ON
 .WORD M,M,M,M ; COMPLEMENT OF VALID EVENT MASKS FOR CLASS
 EVTVAL: .WORD CL\$MAN/100 ; EVENT CLASS 0 - NETWORK MANAGEMENT EVENTS
 .WORD ^C<1777>,^C<0>,^C<0>,^C<0>
 .WORD CL\$SES/100 ; EVENT CLASS 2 - SESSION CONTROL EVENTS
 .WORD ^C<3>,^C<0>,^C<0>,^C<0>
 .WORD CL\$ECL/100 ; EVENT CLASS 3 - ECL EVENTS
 .WORD ^C<7>,^C<0>,^C<0>,^C<0>
 .WORD CL\$TRN/100 ; EVENT CLASS 4 - TRANSPORT EVENTS
 .WORD ^C<17777>,^C<7>,^C<0>,^C<0>
 .WORD CL\$DLL/100 ; EVENT CLASS 5 - DATA LINK EVENTS
 .WORD ^C<17777>,^C<1>,^C<0>,^C<0>
 .WORD CL\$PLL/100 ; EVENT CLASS 6 - PHYSICAL LINK LAYER
 .WORD ^C<77>,^C<0>,^C<0>,^C<0>
 .WORD CL\$ROU/100 ; EVENT CLASS 64. - ROUTING- RSX SPECIFIC
 .WORD ^C<6>,^C<0>,^C<0>,^C<0>
 .WORD CL\$LDN/100 ; EVENT CLASS 68. - LINE DOWN (RSX SPECIFIC)
 .WORD ^C<40000>,^C<0>,^C<0>,^C<0>
 .WORD CL\$ASZ/100 ; EVENT CLASS 69. - ASCII STRING (RSX SPECIFIC)
 .WORD ^C<1>,^C<17>,^C<0>,^C<0>
 .WORD CL\$X2S/100 ; EVENT CLASS 93. - X.25 SERVER
 .WORD ^C<1>,^C<0>,^C<0>,^C<0>
 .WORD CL\$XL3/100 ; EVENT CLASS 94. - X.25 LEVEL 3
 .WORD ^C<1>,^C<0>,^C<0>,^C<0>
 .WORD CL\$PLH/100 ; EVENT CLASS 224. - PLUTO SPECIFIC HRDW EVENT
 .WORD ^C<17>,^C<0>,^C<0>,^C<0>
 .WORD CL\$PAZ/100 ; EVENT CLASS 225. - PLUTO SPECIFIC ASCII EVENT
 .WORD ^C<1401>,^C<0>,^C<0>,^C<0>

```

845
846
847
848
849
850
851
852
853
854
855
856
857 002222
858 002222 010146
859 002224 01J246
860 002226 010346
861 002230 016701 000000G
862 002234 016703 000000G
863 002240 001406
864 002242 012102
865 002244 020062 000000G
866 002250 001404
867 002252 005303
868 002254 003372
869 002 56 000261
870 002260 000405
871 002262 005741
872 002264 166701 000000G
873 002270 010100
874 002272 000241
875 002274 012603
876 002276 012602
877 002300 012601
878 002302

      .SBTTL  FNPID - FIND A PROCESS PDV INDEX
      ;+
      ;**--FNPID-FIND A PROCESS'S PDV INDEX
      ;
      ; INPUTS:
      ;      RO = PDV PROCESS NAME TO LOOK FOR
      ;
      ; OUTPUTS:
      ;      C-BIT = SUCCESS/FAILURE
      ;
      ; -
      FNPID:
      MOV     R1,-(SP)      ; SAVE R1
      MOV     R2,-(SP)      ; SAVE R2
      MOV     R3,-(SP)      ; SAVE R3
      MOV     $PDVTA,R1     ; GET ADDRESS OF PDV INDEX TABLE
      MOV     $PDVNM,R3     ; GET NUMBER OF PDV ENTRIES
      BEQ     15$           ; IF EQ, NONE ???
      MOV     (R1)+,R2       ; GET NEXT PDV ENTRY
      CMP     R0,Z.NAM(R2)   ; IS THIS THE PROCESS WE WANT ?
      BEQ     20$           ; IF EQ, YES
      DEC     R3             ; ELSE, ARE THERE ANY MORE PDV'S ?
      BGT     10$           ; IF GT, YES
      SEC     15$           ; ELSE, INDICATE FAILURE
      BR      30$           ; AND RETURN
      TST     -(R1)          ; BACK UP INDEX TABLE POINTER
      SUB     $PDVTA,R1     ; CALCULATE PDV INDEX
      MOV     R1,RO         ; RETURN PDV INDEX IN ADVERTISED PLACE
      CLC                     ; INDICATE SUCCESS
      30$:    MOV     (SP)+,R3 ; RESTORE REGISTERS
      MOV     (SP)+,R2
      MOV     (SP)+,R1
      RETURN

```


G 13

```

1423      ;
1424      ; LOOPBACK NODE ALREADY EXISTS
1425      ;
1426      004446 012700 000000G      MOV      #$BFR,R0      ; POINT TO I/O BUFFER
1427      004452 016002 000010      MCV      R.ADD(R0),R2      ; GET CHANEL NUMBER
1428      004456 005760 000012      TST      R.FLAG(R0)      ; CHECK IF LOOPBACK
1429      004462 100060      BPL      101$      ; IF PL, NOT A LOOPBACK ... ERROR
1430      004464      CALL      CLRBIT      ; ELSE, CLEAR LOOPBACK BIT
1431      004470      BCS      30$      ; IF CS, ERROR
1432      004472 016760 000044G 000010      MOV      RQB+LO.NAM,R.ADD(R0)      ; STORE NEW CHANNEL NUMBER
1433      004500 052760 100000 000012      BIS      #RF.LOD,R.FLAG(R0)      ; SET LOOPBACK BIT
1434      004506      PUTAD      ; RE-WRITE REMOTE BLOCK
1435      004512 016002 000010      MOV      R.ADD(R0),R2      ; GET NEW CHANNEL NUMBER
1436      004516      CALL      SETBIT      ; AND SET LOOPBACK BIT
1437      004522 000437      BR      30$      ; AND LEAVE
1438
1439      ;
1440      ; CREATE NEW LOOPBACK ENTRY
1441      004524 016702 000044G      10$: MOV      RQB+LO.NAM,R2      ; GET CHANNEL NUMBER
1442      004530      CALL      SETBIT      ; AND SET LOOPBACK BIT
1443      004534 103432      BCS      30$      ; IF CS, ERROR
1444      004536 012701 000014      MOV      #R.LEN,R1      ; GET LENGTH OF REMOTE BLOCK
1445      004542      CALL      $ALPOL      ; TRY TO ALLOCATE A REMOTE BLOCK
1446      004546 103432      BCS      102$      ; IF CS, ALLOCATE FAILURE
1447      004550 010067 000164'      MOV      R0,CURUNM      ; SAVE UNMAPPED ADDRESS
1448      004554 012'01 000000G      MOV      #$BFR,R1      ; ELSE, POINT TO I/O BUFFER
1449      004560 062701 000002      ADD      #R.NAM,R1      ; POINT TO NAME FIELD
1450      004564 012702 000012G      MOV      #RQB+LR.NAM,R2      ; POINT TO INPUT NAME
1451      004570 012221      MOV      (R2)+,(R1)+      ; STORE NAME IN REMOTE BLOCK
1452      004572 012221      MOV      (R2)+,(R1)+      ; ...
1453      004574 012221      MOV      (R2)+,(R1)+      ; ...
1454      004576 162701 000010      SUB      #R.NAM+6,R1      ; POINT TO START OF BLOCK AGAIN
1455      004602 016761 000044G 000010      MOV      RQB+LO.NAM,R.ADD(R1)      ; STORE CHANNEL NO.
1456      004610 052761 100000 000012      BIS      #RF.LOD,R.FLAG(R1)      ; SET LOOPBACK
1457
1458      ;
1459      ; INSERT ENTRY INTO REMOTE LIST
1460      004616 000167 000350      JMP      REMINS      ; INSERT ENTRY INTO REMOTE LIST
1461      004622      30$: RETURN
1462
1463      ;
1464      ; ERROR CONDITIONS
1465      004624      101$: ERRPT$ ERR17,$EROUT      ; INVALID NODE OPTION
1466      004634      102$: ERRPT$ ERR18,$EROUT      ; REMOTE BLOCK ALLOCATION FAILURE
1467      004644      103$: ERRPT$ ERR19,$EROUT      ; NODE NOT IN SYSTEM
1468
1469      .DSABL LSB

```

H 13

```

1894                                     .SBTTL $VERIF - SET VERIFICATION STATE
1895
1896                                     ;+
1897                                     *** - $VERIF - SET VERIFICATION STATE ON/OFF
1898
1899                                     INPUTS:
1900                                     RQB+LO.VER = VERIFICATION STATE (0=OFF/1=ON)
1901
1902                                     OUTPUTS:
1903                                     VERIFICATION STATE IS SET
1904                                     C-BIT = SUCCESS/FAILURE
1905
1906                                     :-
1907
1908 $VERIF::
1909     006410 012700 051516     MOV     #'NS,RO           ; GET NETWORK DEVICE NAME
1910     006410     CALL     FNDDCB           ; FIND THE DCB
1911     006420 103431     BCS     101$         ; IF CS, NOT THERE .. ERROR
1912     006422 016767     MOV     $BFR+D,UCB,$RADDR       ; GET ADDRESS OF UCB
1913     006430     GETRV    ,#U.CW3+2       ; READ IN THE UCB
1914     006442 032767     BIT     #NF.ACC,$BFR+U.CW3     ; IS ACCESS VERIFICATION SUPPORTED ?
1915     006450 001421     BEQ     102$         ; IF EQ, NO .. ERROR
1916     006452 052767     BIS     #NF$ACC,$BFR+U.CW3     ; ASSUME ACCESS VERIFICATION WANTED
1917     006460 032767     BIT     #1,RQB+LO.VER         ; DO WE WANT VERIFICATION ?
1918     006466 001003     BNE     10$          ; IF NE, YES .. OKAY
1919     006470 042767     BIC     #NF$ACC,$BFR+U.CW3     ; ELSE, NO VERIFICATION
1920     006502 100000 000014G 10$: PUTRC    ; REWRITE THE UCB
1921     006502     RETURN
1922
1923                                     ;
1924                                     ; ERROR CONDITIONS
1925
1926     006504 101$: ERRPT$ ERR3,$EROUT       ; XPT PROCESS NOT LOADED
1927     006514 102$: ERRPT$ ERR23,$EROUT     ; ACCESS VERIFICATION NOT SUPPORTED

```

VENA - VNP ENABLE/DISABLE LINES MACRO V05.03b Wednesday 11-Sep-85 12:06 Page 44-1
\$VLOG - PROCESS LOGGING REQUEST

2534 012134
2535
2536

101\$: ERRPT\$ ERR27,\$EROUT

; EVENT LOGGING NOT SUPPORTED

```

3044                                     .SBTTL CKNODE - CHECK VALIDITY OF NODE ENTITY ID
3045
3046                                     ;
3047                                     ; CKNODE - CHECK VALIDITY OF NODE ENTITY ID
3048                                     ;
3049                                     ; INPUTS:
3050                                     ; LO,NM+RQB - NODE ID
3051                                     ; $VFLAG - FLAGS WORD
3052                                     ;
3053                                     ; OUTPUTS:
3054                                     ; REMOTE - REMOTE NODE ADDRESS IF VALID
3055                                     ; CARRY - SUCCESS/FAILURE
3056                                     ;
3057                                     ;
3058                                     ;
3059
3060 014070 012701 000022G CKNODE: MOV #RQB+LO,NM,R1 ; POINT TO NODE ID
3061 014074 032767 000000G 000000G BIT #VF.EAD,$VFLAG ; NODE ADDRESS SPECIFIED?
3062 014102 001403 BEQ 10$ ; BR IF NO
3063 014104 011167 000132' MOV (R1),REMOTE ; SAVE NODE ADDRESS
3064 014110 000406 BR 20$ ; CONTINUE
3065 014112 103405 10$: CALL $FNAM ; IS THERE A REMOTE NODE NAME BLOCK?
3066 014116 016767 000010G 000132' BCS 101$ ; BR IF NO
3067 014120 000241 MOV $BFR+R.ADD,REMOTE ; SAVE NODE ADDRESS
3068 014126 20$: CLC ; INDICATE SUCCESS
3069 014130 30$: RETURN
3070
3071 ; ERROR MESSAGES
3072
3073 014132 101$: ERPT$ ERR40,$EROUT ; ILLEGAL REMOTE NODE PARAMETER

```

```

1353                                     .SBTTL  CHKBIT - CHECK TO SEE IF BIT IS ON
1354
1355                                     ;+
1356                                     ;:
1357                                     ;:  CHKBIT - CHECK TO SEE IF SPECIFIED BIT IS ON
1358                                     ;:
1359                                     ;:  INPUTS:
1360                                     ;:    BITNUM - BIT NUMBER TO CHECK
1361                                     ;:
1362                                     ;:  OUTPUTS:
1363                                     ;:    CONDITION CODE AS DEFINED FOR BITB INSTRUCTION
1364                                     ;:
1365                                     ;:-
1366 007230 012702 000001  CHKBIT: MOV     #1,R2          ; GET A CHECK BIT IN POSITION 0
1367 007234 116701 171744      MOV     BITNUM,R1       ; GET BIT NUMBER
1368 007240 042701 177770      BIC     #177770,R1      ; MAKE IT A SHIFT COUNT
1369 007244 001403              BEQ     20$            ; BR IF BIT POSITION 0
1370 007246 006302      10$:   ASL     R2              ; SHIFT CHECK BIT LEFT
1371 007250 005301      DEC     R1                    ; DECREMENT SHIFT COUNT
1372 007252 003375      BGT     10$                    ; BR IF MORE TO SHIFT
1373 007254 116701 171724      20$:   MOV     BITNUM,R1  ; GET BIT NUMBER
1374 007260 006201      ASR     R1                    ; DIVIDE BY 8 TO GET BYTE INDEX
1375 007262 006201      ASR     R1                    ; ...
1376 007264 006201      ASR     R1                    ; ...
1377 007266 062701 001034'    ADD     #EVTMSK,R1       ; GET ADDRESS OF EVENT MASKS
1378 007272 130211      BITB    R2,(R1)               ; CHECK BIT
1379 007274              RETURN

```

```

2021 014502 026167 000000G 000002G CMP Z,NAM(R1),LR.LIN+RQB ; IS THIS A MATCH?
2022 014510 001045 BNE 250$ ; BR IF NO
2023
2024 ; MATCH ON CONTROLLER NUMBER
2025
2026 014512 032767 000400 000000G BIT #WC,CNT,$QUAL ; WILD CARD CONTROLLER?
2027 014520 001004 BNE 210$ ; BR IF YES, MATCH
2028 014522 126067 000012 000004G CMPB L,CTL(R0),LR.CTL+RQB ; IS THIS A MATCH?
2029 014530 001035 BNE 250$ ; BR IF NO
2030
2031 ; MATCH ON UNIT NUMBER
2032
2033 014532 032767 001000 000000G 210$: BIT #WC,UNT,$QUAL ; WILD CARD UNIT?
2034 014540 001004 BNE 220$ ; BR IF YES, MATCH
2035 014542 126067 000013 000005G CMPB L,UNT(R0),LR.UNT+RQB ; IS THIS A MATCH?
2036 014550 001025 BNE 250$ ; BR IF NO
2037
2038 ; MATCH ON TRIBUTARY NUMBER
2039
2040 014552 116002 000014 220$: MOVB L,NSTA(R0),R2 ; IS THIS A MULTIPOINT LINE?
2041 014556 001443 BEQ 290$ ; BR IF NO
2042 014560 032767 002000 000000G BIT #WC,TRI,$QUAL ; WILD CARD TRIBUTARY?
2043 014566 001020 BNE 260$ ; BR IF YES, MATCH
2044 014570 010246 MOV R2,-(SP) ; USE STACK FOR COUNTER
2045 014572 005002 CLR R2 ; TRIBUTARY NUMBER
2046 014574 010005 MOV R0,R5 ; GET SLT ADDRESS
2047 014576 062705 000022 ADD #L,MPF,R5 ; POINT TO MULTIPOINT FLAG TABLE
2048 014602 126702 000006G 230$: CMPB LR,TRI+RQB,R2 ; IS THIS A MATCH?
2049 014606 001426 BEQ 280$ ; BR IF YES
2050 014610 005202 INC R2 ; INCREMENT STATION NUMBER
2051 014612 062705 000004 ADD #S,LEN,R5 ; POINT TO NEXT TRIBUTARY
2052 014616 005316 DEC (SP) ; DECREMENT NUMBER OF STATIONS
2053 014620 003370 BGT 230$ ; BR IF NO
2054 014622 005726 240$: TST (SP)+ ; CLEAN UP STACK
2055 014624 250$: RESRG <R5> ; RESTORE R5
2056 014626 000653 BR 160$ ; NO MATCH, TRY NEXT LINE
2057
2058 ; FOUND A MATCH - WILD CARD TRIBUTARY
2059
2060 014630 010246 260$: MOV R2,-(SP) ; GET NUMBER OF STATIONS
2061 014632 010005 MOV R0,R5 ; GET SLT ADDRESS
2062 014634 062705 000022 ADD #L,MPF,R5 ; POINT TO MULTIPOINT FLAG TABLE
2063 014640 005002 CLR R2 ; FIRST TRIBUTARY
2064 014642 270$: CALL DISCIR ; DISPLAY STATION INFORMATION
2065 014646 005204 INC R4 ; INDICATE HEADER NOT TO BE DISPLAYED
2066 014650 005202 INC R2 ; INCREMENT STATION NUMBER
2067 014652 062705 000004 ADD #S,LEN,R5 ; POINT TO NEXT TRIBUTARY
2068 014656 005316 DEC (SP) ; DECREMENT NUMBER OF STATIONS
2069 014660 003370 BGT 270$ ; BR IF NO
2070 014662 000757 BR 240$ ; CLEAN UP STACK AND CONTINUE SEARCH
2071
2072 014664 005726 280$: TST (SP)+ ; CLEAN UP STACK
2073 014666 290$: CALL DISCIR ; DISPLAY CIRCUIT INFORMATION
2074 014672 005204 INC R4 ; INDICATE HEADER TO BE DISPLAYED
2075 014674 000753 BR 250$ ; CONTINUE SCAN
2076
2077 ; SHOW KNOWN CIRCUITS

```

```

2570 017600          016767 000002G 161236 90$: GETAD R0,#Y$LEN ; READ IN LINE TABLE
2571 017616          016767 000002G 161352 MOV Y$DPSZ+$BFR,TEMP2 ; GET DEFAULT DATA
2572 017624          005067 161352 CLR TEMP3 ; GET DEFAULT WINDOW
2573 017630          156767 000006G 161344 BISB Y$DWSZ+$BFR,TEMP3 ;
2574 017636          156767 000006G 161344 ERRPT$ XPM$1,MSGOUT ; DISPLAY INFO TO TERMINAL
2575
2576 017646          016767 000004G 161160 MOV Y$MPSZ+$BFR,TEMP1 ; GET MAXIMUM DATA
2577 017654          005067 161202 CLR TEMP2 ; GET MAXIMUM WINDOW
2578 017660          156767 000007G 161174 BISB Y$MWSZ+$BFR,TEMP2 ;
2579 017666          005067 161310 CLR TEMP3 ; GET MAXIMUM CLEARS
2580 017672          156767 000016G 161302 BISB Y$MCLR+$BFR,TEMP3 ;
2581 017700          156767 000016G 161302 ERRPT$ XPM$2,MSGOUT ; DISPLAY INFO TO TERMINAL
2582
2583 017710          005067 161120 CLR TEMP1 ; GET MAXIMUM RESETS
2584 017714          156767 000015G 161112 BISB Y$MRES+$BFR,TEMP1 ;
2585 017722          005067 161134 CLR TEMP2 ; GET MAXIMUM RESTARTS
2586 017726          156767 000014G 161126 BISB Y$MRST+$BFR,TEMP2 ;
2587 017734          005067 161242 CLR TEMP3 ; GET CALL TIMER
2588 017740          156767 000011G 161234 BISB Y$TCAL+$BFR,TEMP3 ;
2589 017746          156767 000011G 161234 ERRPT$ XPM$3,MSGOUT ; DISPLAY INFO TO TERMINAL
2590
2591 017756          005067 161052 CLR TEMP1 ; GET CLEAR TIMER
2592 017762          156767 000013G 161044 BISB Y$TCLR+$BFR,TEMP1 ;
2593 017770          005067 161066 CLR TEMP2 ; GET RESET TIMER
2594 017774          156767 000012G 161060 BISB Y$TRES+$BFR,TEMP2 ;
2595 020002          005067 161174 CLR TEMP3 ; GET RESTART TIMER
2596 020006          156767 000010G 161166 BISB Y$TRST+$BFR,TEMP3 ;
2597 020014          156767 000010G 161166 ERRPT$ XPM$4,MSGOUT ; DISPLAY INFO TO TERMINAL
2598 020024          156767 000010G 161166 RETURN
2599
2600 ; ERRORS
2601 ;
2602 020026          005067 161052 ERMSG$ < PLI process not in system>
2603 020060          005067 161066 ERBLK$ XPER1
2604 ;
2605 ; DISPLAY MESSAGES
2606 ;
2607 020064          005067 161052 ERMSG$ < Network name = %A, Default data = %D, Default window = %D>
2608 020161          005067 161066 ERBLK$ XPM$1,<TEMP1,TEMP2,TEMP3>
2609 ;
2610 020172          005067 161052 ERMSG$ < Maximum data = %D, Maximum window = %D, Maximum clears = %D>
2611 020271          005067 161066 ERBLK$ XPM$2,<TEMP1,TEMP2,TEMP3>
2612 ;
2613 020302          005067 161052 ERMSG$ < Maximum resets = %D, Maximum restarts = %D, Call timer = %D>
2614 020401          005067 161066 ERBLK$ XPM$3,<TEMP1,TEMP2,TEMP3>
2615 ;
2616 020412          005067 161052 ERMSG$ < Clear timer = %D, Reset timer = %D, Restart timer = %D%N>
2617 020506          005067 161066 ERBLK$ XPM$4,<TEMP1,TEMP2,TEMP3>

```

```

3112 .SBTTL NXTDST - Get next destination block ;[TD001]
3113 ;**2
3114
3115 *
3116 *** NXTDST - Get next destination block ;[TD001]
3117 ;[TD001]
3118 INPUTS: ;[TD001]
3119 $BFR - Address of next destination descriptor ;[TD001]
3120 $OPTON - Options word ;[TD001]
3121 LO.DES+RQB - Destination name to look for ;[TD001]
3122 ;**6
3123
3124 OUTPUTS:
3125 CARRY CLEAR: ;[TD001]
3126 $BFR contains next destination block ;[TD001]
3127 R0 - address of destination name ;[TD001]
3128 R1 - length of destination name ;[TD001]
3129 CARRY SET: ;[TD001]
3130 End of destination blocks ;[TD001]
3131
3132 Registers destroyed ;[TD001]
3133
3134 NXTDST: ;[TD001]
3135 CLR R0 ; Presume no specific destination ;[TD001]
3136 BIT #OP$KDS,$OPTON ; Is known destinations specified? ;[TD001]
3137 BNE 10$ ; If so, continue ;[TD001]
3138 MOV #RQB+LO.DES,R0 ; Else look for specific destination ;[TD001]
3139 BIT #PF.XDF,$PFLAG ; Check for extended data format ;[TD001]
3140 BNE 20$ ; Branch if extended ;[TD001]
3141 MOV #D$FLEN+MXCALL+MXDTEA,R1 ; Max length of old format block ;[TD001]
3142 MOV #D$DST,R2 ; Offset of destination name ;[TD001]
3143 MOV #-6,R3 ; Dest name is fixed, 6 characters ;[TD001]
3144 BR 30$ ;[TD001]
3145 MOV #D$GLEN+MXCALL+MXDTEA+MXACCT+<3*MAXID>,R1 ; new max length ;[TD001]
3146 MOV #D$GVL,R2 ; Offset of destination name ;[TD001]
3147 MOV #D$DSL,R3 ; Offset of destination name length ;[TD001]
3148 CALL FNDBLK ; Find matching block ;[TD001]
3149 BCS 99$ ; Branch if not found ;[TD001]
3150 BISB #F.FND,FLAGS ; Signal destination found ;[TD001]
3151 CLC ; Signal block found ;[TD001]
3152 99$: RETURN ;[TD001]

```



```

3671          .SBTTL  DISPRO - DISPLAY PROCESS INFORMATION
3672
3673          ;+
3674          :
3675          : DISPRO - DISPLAY PROCESS INFORMATION
3676          :
3677          : INPUTS:
3678          : R2 = ADDRESS OF PROCESS DESCRIPTOR VECTOR (PDV)
3679          : R1 = 0 IF HEADER IS TO BE DISPLAYED ALONG WITH PROCESS INFORMATION,
3680          :       OTHERWISE DISPLAY ONLY PROCESS INFORMATION
3681          : IF HEADER MESSAGE IS TO BE DISPLAYED:
3682          :     TEMP1 = SECOND ASCIZ STRING OF HEADER MESSAGE
3683          :     TEMP2 = FIRST ASCIZ STRING OF HEADER MESSAGE
3684          :
3685          : OUTPUTS:
3686          : THE PROCESS INFORMATION IS DISPLAYED ON THE USER'S TERMINAL
3687          :
3688          : R3,R5 DESTROYED
3689          : ALL OTHER REGISTERS PRESERVED
3690          :
3691          :-
3692
3693          DISPRO: SAVRG  <R1,R2>          ; SAVE R1 AND R2
3694                   TST   R1              ; PRINT HEADER?
3695                   BNE   10$             ; BR IF NO
3696                   ERRT$  PMSG1,MSGOUT    ; DISPLAY HEADER
3697                   MOV   (SP),R2         ; RESTORE R2
3698
3699          :
3700          : DISPLAY PROCESS NAME
3701          :
3702          : 10$:  MOV   Z.NAM(R2),TEMP3    ; MOVE NAME TO OUTPUT AREA
3703          :       ERRT$  PMSG2,MSGOUT      ; DISPLAY PROCESS NAME
3704          :       MOV   (SP),R2           ; RESTORE R2
3705          :
3706          : DISPLAY PROCESS STATE
3707          :
3708          : 20$:  MOV   #TEMP2,R5         ; STORAGE FOR STATE STRING
3709          :       MOV   #ON,R3            ; ASSUME STATE IS ON
3710          :       TST   Z.PCB(R2)         ; IS PROCESS STATE ON?
3711          :       BNE   20$               ; BR IF YES
3712          :       MOV   #PCLEAR,R3        ; ELSE IT IS CLEAR
3713          :       MOV   (R3)+,(R5)+       ; STORE THE STATE STRING
3714          :       BNE   20$               ;
3715          :       ERRT$  PMSG3,MSGOUT      ; DISPLAY THE PROCESS STATE
3716          :       RESRG  <R2,R1>          ; RESTORE R2 AND R1
3717          :       RETURN                  ; FINISHED
3718
3719          .ENABL  LC
3720
3721          :
3722          : DISPLAY MESSAGES
3723          :
3724          : ERMSG$ <%N%A %A as of %Y%N>
3725          : ERBLK$ PMSG1,<TEMP2,TEMP1>
3726
3727          : ERMSG$ <Process = %R%N>
3728          : ERBLK$ PMSG2,<TEMP3>

```

```

4266 032062 001403 BEQ 530$ ; IF EQ, YES - READ THEM
4267 032064 062701 000046 ADD #T$LEN,R1 ; ELSE, POINT TO NEXT COUNTERS
4268 032070 000761 BR 520$ ; AND KEEP LOOKING
4269 032072 016767 000030G 146734 530$: MOV T$T5+$BFR,TEMP1 ; GET HELLO TIMER
4270 032100 016767 000032G 146754 MOV T$T6+$BFR,TEMP2 ; GET LISTEN TIMER
4271 032106 ERRPT$ MSG4,MSGOUT ; DISPLAY HELLO AND LISTEN TIMER
4272
4273 032116 540$: ERRPT$ MSG12,MSGOUT ; MAKE OUTPUT LOOK NICE
4274 032126 RESRG <R5,R4,R3,R2,R1,R0>
4275 032142 RETURN
4276
4277 .ENABL LC
4278
4279 ;
4280 ; DISPLAY MESSAGES
4281 ;
4282
4283 032144 ERMSG$ <%N%A %A as of %Y%N>
4284 032166 ERBLK$ MSG1,<TEMP2,TEMP1>
4285
4286 032176 ERMSG$ <%NCircuit = %A%N>
4287 032216 ERBLK$ MSG2,<TEMP3>
4288
4289 032224 ERMSG$ < Hello timer = %D, Listen timer = %D>
4290 032272 ERBLK$ MSG4,<TEMP1,TEMP2>
4291
4292 032302 ERMSG$ < State = %A, Loopback Name = %A>
4293 032343 ERBLK$ MSG5,<TEMP1,TEMP2>
4294
4295 032352 ERMSG$ < State = %A, Service = %A>
4296 032405 ERBLK$ MSG6,<TEMP1,TEMP2>
4297
4298 032414 ERMSG$ < Owner = %A>
4299 032431 ERBLK$ MSG7,<TEMP3>
4300
4301 032436 ERMSG$ < Cost = %E, Type = %A, Tributary = %B>
4302 032505 ERBLK$ MSG8,<TEMP1,TEMP2,TEMP3>
4303
4304 032516 ERMSG$ < Cost = %E, Type = %A>
4305 032545 ERBLK$ MSG9,<TEMP1,TEMP2>
4306
4307
4308 032554 ERMSG$ < Cost = %E>
4309 032570 ERBLK$ MSG11,<TEMP1>
4310
4311 032576 ERMSG$ < >
4312 032577 ERBLK$ MSG12
4313
4314 .DSABL LC

```

```

4761                                     .SBTTL TSKSCH - SEARCH TCB'S FOR SPECIFIED TASK
4762
4763                                     ;+
4764                                     ; TSKSCH - SEARCH TCB'S FOR SPECIFIED TASK
4765                                     ;
4766                                     ; INPUTS:
4767                                     ; R2 - POINTER TO RAD50 TASK NAME
4768                                     ;
4769                                     ; OUTPUTS:
4770                                     ; $RADDR - ADDRESS OF TCB
4771                                     ; $BFR - IMAGE OF TCB
4772                                     ;
4773                                     ; R5 DESTROYED
4774                                     ;
4775                                     ; -
4776
4777
4778 035052 TSKSCH: GETRV .TSKHD, TLGTH ; READ IN LISTHEAD
4779 035072 012705 000000G MOV #BFR, R5 ; POINT AT BUFFER BASE
4780 035076 011565 000030 MOV (R5), T.TCBL(R5) ; INITIALIZE FOR LOOP
4781
4782 035102 016567 000030 000000G 10$: MOV T.TCBL(R5), $RADDR ; SET TCB ADDRESS
4783 035110 000261 SEC ; ASSUME ERROR
4784 035112 001411 BEQ 30$ ; NULL TASK ( END OF LIST )
4785 035114 GETRV ; READ IN TCB
4786 035120 021265 000006 CMP (R2), T.NAM(R5) ; MUST MATCH BOTH
4787 035124 001366 BNE 10$
4788 035126 026265 000002 000010 CMP 2(R2), T.NAM+2(R5) ; HALVES OF NAME
4789 035134 001362 BNE 10$ ; C-BIT IS CLEAR
4790 035136 30$: RETURN

```

VDIS - VNP SHOW COMMANDS
Symbol table

MACRO V05.03b Monday 15-Jul-85 12:14 ^{H 8} Page 66-7

Work file reads: 280
Work file writes: 264
Size of work file: 45156 Words (177 Pages)
Size of core pool: 17608 Words (67 Pages)
Operating system: RSX-11M/PLUS

Elapsed time: 00:04:16.21

DB2:VDIS.T34,[132,134]VDIS/CR/-SP=DB2:[1,1]RSXMCM.SML/ML,[130,110]NETLIB/ML,[130,10]RSXMCM/PA:1,[132,10]VDIS

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 13 H 9
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
OP\$X5S	= 000004	#5-99 29-2288
OP\$X9S	= 000010	#5-99 29-2289
OSPEC	000055 R	#6-124 9-410
O.FLG	000003	48-3470 48-3494
O.LEN	000012	9-387
O.MXC	000004	48-3488 48-3499
O.NAM	000006	48-3463 48-3464
O.TYP	000002	9-416 48-3458 48-3480
PACTIV	000175 R	#6-155 11-584
PCLEAR	000246 R	#6-162 50-3711 53-4134 55-4499
PERIOD	= 000056	#7-302 19-1133 21-1259 49-3620 57-4660
PERR1	003444 R	11-629 #11-641
PF.XDF	= ***** GX	27-1952 27-2005 27-2094 34-2546 38-2821 40-2938 40-3028 41-3138 54-4341
PF.XGA	= ***** GX	33-2426
PKNOWN	000216 R	#6-156 11-614
PLIPDV	001030 R	#6-243 *34-2531 35-2654
PMSG1	026650 R	50-3696 #50-3724
PMSG2	026676 R	50-3702 #50-3727
PMSG3	026722 R	50-3714 #50-3730
PREVNT	= 001205 R	#7-318 *19-1145 *19-1152 *19-1161 23-1346
PSPEC	000236 R	#6-157 11-599
PVMSG1	032756 R	54-4336 #54-4363
PVMSG2	033006 R	54-4352 #54-4366
PVMSG3	033044 R	54-4353 #54-4369
PVMSG4	033070 R	54-4355 #54-4372
P.LEN	= 000154	53-4080
P.TRIB	000014	53-4081
QF\$ACT	= 000001	#5-99 12-691 13-830
QF\$ALL	= 000002	#5-99
QF\$KNO	= 000004	#5-99 10-470 11-579 27-1931 28-2163
QF\$LOO	= 000010	#5-99 26-1447
Q\$CTIM	000000	42-3181
Q\$LEN	000016	42-3180
Q\$MXVC	000002	42-3182
Q.IOFN	= 000002	6-267 6-268
RCVLNG	= 000044	#7-307 39-2880 59-4733
RF.LOO	= 100000	#5-94
ROUTNG	= 000416 R	#6-189 26-1671
RQB	= ***** GX	9-415 10-495 10-525 11-602 12-675 *12-681 17-981 17-984 26-1526
		27-2004 27-2021 27-2028 27-2035 27-2048 28-2194 28-2201 28-2208 32-2406
		53-2435 36-2711 36-2713 36-2715 38-2820 41-3137
RX.NXM	= 000020	#5-88
RX.OVR	= 000010	#5-88
R\$ACC	000064	33-2470
R\$ACL	000063	33-2468
R\$DAL	000013	31-2344
R\$DTE	000014	31-2345
R\$LEN	000024	32-2407
R\$NAM	000002	31-2334 32-2408
R\$NML	000002	33-2438
R\$NOD	000023	33-2452
R\$NWK	000003	33-2437

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 26 H 10
 MACRO CROSS REFERENCE CREF 04.00

MACRO NAME	REFERENCES
.CNB	#5-88
.CNW	#5-88
.CORE	#5-88
.CSR	#5-88 5-88 5-88
.CTIM	#5-88
.DPRB	#5-88
.DPRW	#5-88
.DVCHA	#5-88 5-88
.INT	#5-88 5-88 5-88
.INT1	#5-88
.INT2	#5-88
.INT3	#5-88
.LFLHD	#5-88
.LIBR	#5-88
.LINKS	#5-88
.LSTHD	#5-88 5-88
.LTAB	#5-88
.MPLHD	#5-88 5-88
.MXPTB	#5-88
.PECHA	#5-88
.POOL	#5-88
.PRI	#5-88 5-88 5-88
.SCOM	#5-88
.SECSR	#5-88
.SLNB	#5-88 5-88
.SLNW	#5-88 5-88
.STNB	#5-88 5-88
.STNW	#5-88
.TIME	#5-88
.UNB	#5-88 5-88
.UNW	#5-88
.VFY	#5-88
.X2CHB	#5-88
.X2CHW	#5-88
.X3CHB	#5-88
.X3CHW	#5-88

```

000200 177777
363
364 000202 000342 .WORD CL$PRT/100 ; EVENT CLASS 226. - PLUTO SPECIFIC ROUTER EVENT
365 000204 177776 177777 177777 .WORD ^C<1>,<C<0>,<C<0>,<C<0>
000212 177777
366
367 000214 000350 .WORD CL$SGE/100 ; EVENT CLASS 232. SNA specific - General gateway Events
368 000216 000000 177777 177777 .WORD ^C<177777>,<C<0>,<C<0>,<C<0>
000224 177777
369
370 000226 000351 .WORD CL$SSE/100 ; EVENT CLASS 233. SNA specific - gateway Server Events
371 000230 000000 000000 177777 .WORD ^C<177777>,<C<177777>,<C<0>,<C<0>
000236 177777
372
373
374 000020 EVTVSZ=<.-EVTVAL>/12
375
376 000240 177777 .WORD -1 ; END OF TABLE

```

```

880 .SBTTL FNDLIN - FIND THE DDCMP LINE TABLE FOR A LINE
881 .SBTTL FNDTRI - FIND THE TRIBUTARY TABLE FOR A LINE
882
883 **FNDLIN-FIND THE DDCMP LINE TABLE FOR A LINE
884 **FNDTRI-FIND THE DDCMP TRIBUTARY TABLE FOR A LINE
885
886 THIS ROUTINE IS CALLED TO FIND THE DDCMP LINE/TRIBUTARY TABLE FOR A LINE.
887
888 INPUTS:
889 $SLTA = SLT ADDRESS
890 $TRIB = TRIBUTARY NUMBER
891
892 OUTPUTS:
893 C-BIT = ERROR INDICATOR
894 Z-BIT = SEARCH SUCCESS/FAILURE INDICATOR
895 R0-R2 = DESTROYED
896
897
898 .ENABL LSB
899 FNDLIN: CLR -(SP) ; INDICATE THAT A LINE TABLE IS NEEDED
900 002304 005046 BR 5$ ; AND CONTINUE
901 002306 000402
902 002310 FNDTRI: MOV #1, -(SP) ; INDICATE THE TRIBUTARY TABLE IS NEEDED
903 002310 012746 000001 $SLTA, R0 ; GET SLT ADDRESS
904 002314 016700 000000G 5$: BIT #LF.RDY, L.FIG(R0) ; IS THIS LINE READY ?
905 002320 032760 000000G 000000G BEQ 101$ ; IF EQ, NOT READY
906 002326 001466 MOVBL L.DLC(R0), R2 ; ELSE, GET DLC PDV INDEX
907 002330 116002 000000G ADD $PDVTA, R2 ; POINT INTO PDV MAPPING TABLE
908 002334 066702 000000G MOV (R2), R2 ; POINT TO PDV
909 002340 011202 CMP #*RDCP, Z.NAM(R2) ; IS THIS THE DDCMP PROCESS ?
910 002342 022762 014610 000000G BNE 30$ ; IF NE, NO .. SKIP IT
911 002350 001051 MOV L.DLS(R0), R1 ; GET THE DCP VIRTUAL ADDRESS
912 002352 016001 000000G CLR $RBLCK ; ASSUME UNMAPPED SYSTEM OR DSR ALLOC.
913 002356 005067 000000G TST .MGMGE ; IS THIS A MAPPED SYSTEM ?
914 002362 005767 000000G BMI 10$ ; IF MI, NO .. UNMAPPED
915 002366 100406 CMP #120000, R1 ; ELSE, IS THE LINE TABLE MAPPED ?
916 002370 022701 120000 BHI 10$ ; IF HI, NO
917 002374 101003 MOV L.DLM(R0), $RBLCK ; ELSE, MAP TO THE DCP LINE TABLE
918 002376 116067 000000G 000000G 10$: GETAD R1, #L.PLL+4 ; READ IN THE DCP LINE TABLE
919 00240: TST (SP)+ ; DO WE WANT THE TRIBUTARY TABLE ?
920 002422 005726 BEQ 40$ ; IF EQ, NO .. LEAVE NOW
921 002424 001424 MOV #5BFR, R2 ; POINT TO I/O BUFFER
922 002426 012702 000000G ADD #L.PLL, R2 ; POINT TO THE POLLING LISTHEAD
923 002432 062702 000012 20$: MOV (R2), R2 ; GET THE NEXT ENTRY
924 002436 011202 BEQ 102$ ; IF EQ, NOT HERE - ??
925 002440 001426 GETAD R2, #S.DLCF+2 ; ASSUME $RBLCK ALREADY SET UP
926 002442 000035G 000000G CMPE $BFR+S.STLN, $TRIB ; READ IN THE TRIBUTARY INFO
927 002460 126767 000035G 000000G BNE 20$ ; IS THIS THE TRIBUTARY ?
928 002466 001363 CLZ ; IF NE, NO .. KEEP LOOKING
929 002470 000244 BR 50$ ; ELSE, INDICATE SUCCESS
930 002472 000402 BR 50$ ; AND RETURN
931 002474 005726 30$: TST (SP)+ ; POP STACK
932 002476 000264 40$: SEZ ; INDICATE FAILURE
933 002500 000241 50$: CLC ; INDICATE NO ERROR DETECTED
934 002502
935
936 ;

```



```

1471                                     .SBTTL $VREM - SET REMOTE NODE BLOCK
1472
1473                                     ;*
1474                                     *** - $VREM - SET REMOTE NODE BLOCK
1475                                     :
1476                                     INPUTS:
1477                                     RQB+LR.ADD = NODE ADDRESS/NAME
1478                                     RQB+LO.NAM = NODE NAME
1479                                     :
1480                                     OUTPUTS:
1481                                     RO-R2 = DESTROYED
1482                                     :
1483                                     -
1484                                     .ENABL LSB
1485 004654 $VREM:: BIT #CM$SET,$CMAND ; IS THIS A SET COMMAND ?
1486 004654 032767 000000G 000000G BNE 5$ ; IF NE, YES
1487
1488                                     ;
1489                                     CLEAR REQUEST
1490 004664 032767 000000G 000000G BIT #VF.ADD,$VFLAG ; WAS THE NODE ADDRESS GIVEN ?
1491 004672 001012 BNE 3$ ; IF NE, YES
1492 004674 012701 000012G MOV #RQB+LR.NAM,R1 ; GET ADDRESS OF NODE NAME
1493 004700 CALL $FNAM ; TRY TO FIND THE NODE BLOCK
1494 004704 103422 1$: BCS 6$ ; IF CS, NOT FOUND .. ERROR
1495 004706 005767 000012G TST $BFR+R.FLAG ; IS THIS A REMOTE ?
1496 004712 100437 BMI 15$ ; IF MI, NO .. ERROR
1497 004714 000167 177370 JMP REMNOD ; ELSE, REMOVE NODE ENTRY
1498 004720 016701 000012G 3$: MOV RQB+LR.ADD,R1 ; GET NODE ADDRESS
1499 004724 CALL $FNADD ; TRY TO FIND THE NODE BLOCK
1500 004730 000765 BR 1$ ; AND CONTINUE
1501
1502                                     ;
1503                                     SET REQUEST
1504 004732 032767 000000G 000000G 5$: BIT #VF.ADD,$VFLAG ; WAS THE NODE ADDRESS GIVEN ?
1505 004740 001026 BNE 20$ ; IF NE, YES
1506 004742 012701 000012G MOV #RQB+LR.NAM,R1 ; GET ADDRESS OF REMOTE NODE NAME
1507 004746 CALL $FNAM ; TRY TO FIND THE NODE BLOCK
1508 004752 103420 6$: BCS 18$ ; IF CS, NOT FOUND .. ERROR
1509 004754 005767 000012G TST $BFR+R.FLAG ; IS THIS A REMOTE BLOCK ?
1510 004760 100415 BMI 18$ ; IF MI, NO .. ERROR
1511
1512 004762 012700 000000G 10$: MOV #SBFR,RO ; POINT AT I/O BUFFER
1513 004766 062700 000002 ADD #R.NAM,RO ; POINT TO NAME FIELD
1514 004772 012701 000044G MOV #RQB+LO.NAM,R1 ; POINT TO NEW NODE NAME
1515 004776 012120 MOV (R1)+,(RO)+ ; STORE NEW NODE NAME
1516 005000 012120 MOV (R1)+,(RO)+ ; ...
1517 005002 012120 MOV (R1)+,(RO)+ ; ...
1518 005004 PUTAD ; RE-WRITE REMOTE BLOCK
1519 005010 000533 BR 40$ ; AND LEAVE
1520 005012 000551 15$: BR 101$ ; CONTINUE
1521 005014 000560 18$: BR 103$ ; CONTINUE
1522
1523                                     ;
1524                                     REMOTE NODE ADDRESS WAS GIVEN
1525 005016 005002 20$: CLR R2 ; INITIALIZE DUPLICATE ENTRY FLAG
1526 005020 012701 000044G MOV #RQB+LO.NAM,R1 ; SEE IF DUPLICATE NODE NAME
1527 005024 CALL $FNAM ; ...

```

```

1928 .SBTTL $VACP - MOUNT/DISMOUNT THE ACP
1929
1930 **-$VACP- MOUNT/DISMOUNT ACP
1931
1932 INPUTS:
1933 $VFLAG = TRANSITORY FLAGS
1934
1935 OUTPUTS:
1936 ACP IS MOUNTED/DISMOUNTED OR ERROR CONDITION IS REPORTED
1937
1938
1939 $VACP::
1940 006524 032767 000000C 000000G BIT #VF.PEM!VF.DLX,$VFLAG ; IS THIS A PROTOCOL EMULATOR COMMAND
1941 ; OR DLX ONLY ?
1942 006532 001032 BNE 5$ ; IF NE, YES .. SKIP XPT CHECK
1943 006534 CALL CHKXPT ; CHECK XPT CONTEXT
1944 006540 103002 BCC 2$ ; BR IF OK
1945 006542 CALLR 100$ ; ...
1946
1947 006546 032767 000000G 000000G 2$: BIT #VF.SS,$VFLAG ; S-SYSTEM ?
1948 006554 001002 BNE 3$ ; IF YES - BRANCH
1949 006556 CALLR 112$ ; ELSE ERROR
1950
1951 006562 010167 000072' 3$: MOV R1,PDVX ; SAVE PDV INDEX
1952 006566 012700 000000' MOV #ACPNAM,R0 ; POINT AT RAD50 ACP NAME
1953 006572 CALL TSKSCH ; GET TCB ADDRESS
1954 006576 103002 BCC 4$ ; BR IF FOUND IT
1955 006600 000167 000536 JMP 103$ ; TASK NOT IN SYSTEM !
1956
1957 006604 4$: .IF NDF R$MPL ;[MP01]
1958 BIT #T2.FXD,T.ST2+$BFR ; IS TASK FIXED
1959 006604 032767 002000 000034G BNE 5$ ; BR IF YES
1960 006612 001002 .ENDC ;[MP01]
1961
1962 CALLR 105$ ; ELSE ERROR
1963 006614
1964
1965 006620 012700 000014' 5$: MOV #NT11S,R0 ; ASSUME 11S
1966 006624 CALL TSKSCH ; GET TCB ADDRESS
1967 006630 103002 BCC 12$ ; BR IF FOUND IT
1968 006632 000167 000540 JMP 107$ ; ELSE TASK NOT INSTALLED
1969 006636 12$:
1970 .IF NDF R$MPL ;[MP01]
1971 006636 032767 002000 000034G BIT #T2.FXD,T.ST2+$BFR ; IS TASK FIXED
1972 006644 001002 BNE 15$ ; BR IF YES
1973 .ENDC ;[MP01]
1974
1975 CALLR 108$ ; ELSE ERROR
1976
1977 ; QUEUE SEND DATA TO NTINIT
1978
1979 006652 016705 000000G 15$: MOV $RADDR,R5 ; TCB ADDRESS
1980 006656 012701 000044 MOV #36..R1 ; SIZE OF SEND DATA I/O PACKET
1981
1982 ; CHECK IF MOUNT OR DISMOUNT
1983
1984 006662 032767 000000C 000000G BIT #VF.ON!VF.DLX,$VFLAG ; MOUNT, OR DLX ONLY PASS TO STRT TIMERS

```

```

2538 .SBTTL CKEVE - CHECK FOR VALID EVENT PARAMETER
2539
2540 +
2541 CKEVE - CHECK FOR VALID EVENT PARAMETER
2542
2543 INPUTS:
2544 $OPTON - OPTIONS WORD
2545 $QUAL - COMMAND QUALIFIER WORD
2546 RQB+LO.CLS - EVENT CLASS
2547 RQB+LO.EVT - EVENT FILTER MASK
2548
2549 OUTPUTS:
2550 CARRY CLEAR:
2551 VALID EVENT FILTER MASK
2552 CARRY SET:
2553 INVALID EVENT FILTER MASK
2554
2555 -
2556 CKEVE: CLC ; ASSUME SUCCESS
2557 012144 000241 BIT #OP$KEV,$OPTON ; IS THIS FOR KNOWN EVENTS?
2558 012146 032767 001000 000000G BNE 30$ ; BR IF YES - VALID FILTER MASK
2559 012154 001017 MOV #RQB+LO.EVT,R2 ; POINT TO EVENT MASK
2560 012156 012702 000004G BIT #WC.EVT,$QUAL ; IS THIS FOR WILD CARD EVENTS?
2561 012162 032767 004000 000000G BEQ 20$ ; BR IF NO
2562 012170 001407 MOV R2,R0 ; RETRIEVE POINTER TO EVENT MASK
2563 012172 010200 MOV #8,R1 ; EVENT MASK IS 8. BYTES
2564 012174 012701 000010 10$: MOV #377,(R0)+ ; ELSE SET UP MASK FOR WILD CARD EVENTS
2565 012200 112720 000377 DEC R1 ; MORE TO STORE?
2566 012204 005301 BGT 10$ ; BR IF YES
2567 012206 003374 20$: CALL PROEVT ; PROCESS EVENT MASK
2568 012210 30$: RETURN
2569 012214
2570
2571 +
2572 LINFLT - CLEAR LINE SPECIFIED EVENTS
2573
2574 INPUTS:
2575 $FILHD - FILTER BLOCK LISTHEAD
2576
2577 OUTPUTS:
2578 LINE EVENTS CLEARED
2579
2580 -
2581 LINFLT: CALL $SAVAL ; SAVE REG'S
2582 012216 SWITCHO ; RESET BUFFERS
2583 012222 SWITCHI
2584 012230 MOV #FILHD,R0 ; GET FILTER BLOCK LISTHEAD
2585 012236 012700 000000G MOV (R0),$BFR ; GET FIRST BLOCK
2586 012242 011067 000000G 1$: MOV $BFR,R0 ; GET FIRST BLOCK AGAIN
2587 012246 016700 000000G BNE 2$ ; IF MORE - BRANCH
2588 012252 001001
2589
2590 RETURN
2591
2592 2$: CEACCS R0,R1 ; MAP TO FILTER BLOCK
2593 012256 GETAD R1,#F.LEN ; READ FILTER BLOCK
2594 012304 016767 000002G 000000G MOV F.CLS+$BFR,$RSIZE ; SET UP LENGTH

```

```

3075                                     .SBTTL CHSTA - CHANGE LOGGING STATE
3076
3077                                     ;+
3078                                     ; CHSTA - CHANGE LOGGING STATE
3079                                     ;
3080                                     ; INPUTS:
3081                                     ;   ROB - REQUEST BLOCK FOR SET LOGGING COMMAND
3082                                     ;
3083                                     ; OUTPUTS:
3084                                     ;   SPECIFIED LOGGING STATE IS CHANGED
3085                                     ;
3086                                     ; -
3087
3088
3089 014142 012702 000007 CHSTA: MOV #FF.CON!FF.FIL!FF.MON,R2 ; ASSUME KNOWN LOGGING
3090 014146 032767 000004 000000G BIT #OF$KNO,$QUAL ; IS THIS FOR KNOWN LOGGING?
3091 014154 001016 BNE 10$ ; BR IF YES
3092 014156 012702 000001 MOV #FF.CON,R2 ; ASSUME LOGGING CONSOLE
3093 014162 032767 000400 000000G BIT #OP$CON,$OPTON ; IS THIS FOR THE LOGGING CONSOLE?
3094 014170 001010 BNE 10$ ; BR IF YES
3095 014172 012702 000002 MOV #FF.FIL,R2 ; ASSUME LOGGING FILE
3096 014176 032767 000200 000000G BIT #OP$FIL,$OPTON ; IS THIS FOR LOGGING FILE?
3097 014204 001002 BNE 10$ ; BR IF NO
3098 014206 012702 000004 MOV #FF.MON,R2 ; ELSE MUST BE LOGGING MONITOR
3099
3100 014212 032767 000000G 000000G 10$: BIT #VF.ON,$VFLAG ; SET STATE ON?
3101 014220 001403 BEQ 20$ ; BR IF NO
3102 014222 050267 000000G BIS R2,$LGSTT ; SET LOGGING STATE ON
3103 014226 000402 BR 30$ ; AND CONTINUE
3104 014230 040267 000000G 20$: BIC R2,$LGSTT ; SET LOGGING STATE OFF
3105
3106 014234 000241 30$: CLC ; BE SURE CARRY IS CLEAR
3107 014236 RETURN

```

```

1381 .SBTTL FNDADD - FIND REMOTE NODE BY ADDRESS
1382
1383 ;+
1384 FNDADD - FIND REMOTE NODE BY ADDRESS
1385
1386 INPUTS:
1387 R1 - NODE ADDRESS
1388
1389 OUTPUTS:
1390 C-BIT - SUCCESS/FAILURE
1391 R0 - ADDRESS OF REMOTE BLOCK IF FOUND
1392 R1 - PRESERVED
1393
1394
1395
1396 FNDADD: SWTCHI #SHMBUF ; SET UP INPUT BUFFER BUFFER
1397 GETRV $DECPT,#D$END ; READ DECNET HOME BLOCK
1398 MOV #SHMBUF+D$RNN+2,R0 ; STORE REMOTE LISTHEAD
1399
1400 SWTCHI #TEMP1 ; USE TEMP1 FOR BUFFER
1401 MOV (R0),TEMP1 ; GET FIRST IN LIST
1402 MOV R1,-(SP) ; SAVE NODE ADDRESS ON STACK
1403 10$: MOV R0,R1 ; SAVE ADDRESS OF PREVIOUS
1404 MOV TEMP1,R0 ; GET NEXT REMOTE IN LIST
1405 SEC ; ASSUME FAILURE
1406 BEQ 30$ ; IF EQ, DONE
1407 CEACCS R0 ; CONVERT TO MAPPED ADDRESS
1408 20$: GETAD R0,#R.LEN ; READ REMOTE INTO BUFFER
1409 CMP (SP),TEMP1+R.ADD ; IS THIS THE RIGHT REMOTE?
1410 BNE 10$ ; BR IF NO
1411 CLC
1412 30$: SWTCHI ; RETURN TO DEFAULT BUFFER
1413 MOV (SP)+,R1 ; RESTORE R1
1414 RETURN
  
```

```

2078
2079 014676 012700 000540' 300$: MOV #CKNOWN,R0 : GET STRING FOR HEADER MESSAGE
2080 014702 112021 310$: MOVB (R0)+,(R1)+ : STORE STRING
2081 014704 001376 BNE 310$ :
2082 014706 005004 CLR R4 : INDICATE HEADER TO BE DISPLAYED
2083 014710 320$: CALL NXCIR : GET NEXT LINE IN SYSTEM
2084 014714 103034 BCC 350$ : BR IF END OF TABLE
2085 :
2086 : DISPLAY KNOWN PSI PERMANENT CIRCUITS : [TD001]
2087 : : [TD001]
2088 014716 016700 000000G MOV $PSIPT,R0 : IS THIS A PSI SYSTEM? : [TD001]
2089 014722 001466 BEQ 390$ : BRANCH IF NO - DONE : [TD001]
2090 014724 062700 000006 ADD #H$PVC,R0 : ADDRESS PVC LISTHEAD : [TD001]
2091 014730 GETRV R0,#2 : READ PVC LISTHEAD : [TD001]
2092 014746 005000 CLR R0 : INDICATE NO MATCH CRITERIA : [TD001]
2093 014750 012701 000016 MOV #C$LEN,R1 : LENGTH OF OLD-FORMAT PVC BLOCK : [TD001]
2094 014754 032767 000000G 000000G BIT #PF.XDF,$PFLAG : CHECK FOR EXTENDED DATA FORMAT : [TD001]
2095 014762 001402 BEQ 340$ : IF NONE, THESE PARAMETERS SUFFICE : [TD001]
2096 014764 062701 000020 ADD #MAXID,R1 : OTHERWISE PVC BLOCK MAY BE LONGER : [TD001]
2097 014770 340$: CALL FNDBLK : FIND NEXT PVC : [TD001]
2098 014774 103441 BCS 390$ : BRANCH IF END OF LIST : [TD001]
2099 014776 CALL DISPVC : DISPLAY PVC INFO : [TD001]
2100 015002 005204 INC R4 : UPDATE HEAD ? DISPLAY FLAG : [TD001]
2101 015004 000760 BR 330$ : GET NEXT F C NAME BLOCK : [TD001]
2102 :
2103 015006 032710 000400 350$: BIT #LF.BRO,(R0) : ETHERNET CIRCUIT ? :
2104 015012 001024 BNE 370$ : IF YES - BRANCH :
2105 015014 116002 000014 MOVB L,NSTA(R0),R2 : MULTIPOINT LINE? :
2106 015020 001421 BEQ 370$ : BR IF NO :
2107 015022 SAVRG <R5> : SAVE R5 :
2108 015024 010246 MOV R2,-(SP) : USE STACK AS COUNTER :
2109 015026 010005 MOV R0,R5 : GET SLT ADDRESS :
2110 015030 062705 000022 ADD #L.MPF,R5 : POINT TO MULTIPOINT FLAGS TABLE :
2111 015034 005002 CLR R2 : STATION NUMBER :
2112 015036 360$: CALL DISCIR : DISPLAY TRIBUTARY INFORMATION :
2113 015042 005204 INC R4 : INDICATE HEADER NOT TO BE DISPLAYED :
2114 015044 005202 INC R2 : NEXT STATION NUMBER :
2115 015046 062705 000004 ADD #S.LEN,R5 : POINT TO NEXT STATION :
2116 015052 020216 CMP R2,(SP) : ALL STATIONS DISPLAYED? :
2117 015054 001370 BNE 360$ : BR IF NO :
2118 015056 005726 TST (SP)+ : CLEAR STACK :
2119 015060 RESTRG <R5> : RESTORE R5 :
2120 015062 000712 BR 320$ : GET NEXT LINE :
2121 015064 370$: CALL DISCIR : DISPLAY CIRCUIT INFORMATION :
2122 015070 005204 INC R4 : INDICATE HEADER NOT TO BE DISPLAYED :
2123 015072 000706 BR 320$ : GET NEXT LINE :
2124 015074 005704 380$: TST R4 : WAS THE SPECIFIED LINE FOUND? :
2125 015076 001404 BEQ 400$ : BR IF NO :
2126 :
2127 015100 390$: DIR$ #TIDET : DETACH TERMINAL :
2128 015106 RETURN : FINISHED :
2129 :
2130 015110 400$: DIR$ #TIDET : DETACH TERMINAL :
2131 015116 ERRPT$ CIERR1,$EROUT : LINE NOT IN SYSTEM :
2132 :
2133 :
2134 : ERROR MESSAGES :

```

***-1

```

2619 .SBTTL DMDTE - SHOW MODULE X25-PROTOCOL DTES
2620
2621 ;+
2622 DMDTE - SHOW MODULE X25-PROTOCOL WITH DTE QUALIFIER
2623
2624 INPUTS:
2625 $OPTON - OPTIONS WORD
2626 LD.DTE+ROB - SPECIFIED DTE ADDRESS
2627
2628 OUTPUTS:
2629 DTE INFORMATION DISPLAYED ON USER'S TERMINAL
2630
2631 -
2632
2633 DMDTE: MOV $PSIPT,RO ; POINT TO PSI HOME BLOCK
2634 ADD #H$LDTE,RO ; POINT TO DTE LISTHEAD
2635 GETRV RO,#2 ; READ IN LISTHEAD
2636 10$: CALL NXIDTE ; GET NEXT DTE BLOCK TO OPERATE ON
2637 BCS 40$ ; BR IF DONE
2638 MOV #SBFR+L$DTEA,R1 ; POINT TO DTE ADDRESS
2639 MOVB $BFR+L$DTEL,R2 ; GET NUMBER OF DIGITS IN ADDRESS
2640 MOV #TEMP1,R3 ; POINT TO OUTPUT BUFFER
2641 CALL BCDASC ; CONVERT TO ASCIZ
2642 ERRT$ DTMS1,MSGOUT ; DISPLAY DTE ADDRESS
2643
2644 MOV #OFF,RO ; ASSUME DTE STATE IS OFF
2645 CMPB #LN$OFF,L$NMST+$BFR ; IS DTE STATE OFF?
2646 BEQ 20$ ; BR IF YES
2647 MOV #ON,RO ; ELSE STATE MUST BE ON
2648 20$: MOV #TEMP1,R1 ; POINT TO STORAGE OF STATE STRING
2649 30$: MOVB (RO)+,(R1)+ ; STORE ASCIZ STRING
2650 BNE 30$ ; ...
2651
2652 MOVB L$LLCH+$BFR,RO ; GET CHANNEL NUMBER
2653 ASL RO ; CONVERT TO WORD INDEX
2654 MOV PLIPDV,R1 ; GET PLI'S PDV ADDRESS
2655 ADD R1,RO ; POINT AT CHANNEL IN PDV
2656 MOVB Z$MAP(RO),RO ; GET SLN & TRIBUTARY
2657 ASL RO ; CONVERT TO WORD INDEX
2658 ADD $SLTMA,RO ; POINT AT SLT ADDRESS
2659 MOV (RO),RO ; POINT AT SLT
2660 MOVB L$DDM(RO),R1 ; GET DDM PDV INDEX
2661 ADD $PDVTA,R1 ; POINT AT PDV ADDRESS
2662 MOV (R1),R1 ; POINT AT DDM PDV
2663 CALL FMTLIN ; FORMAT THE LINE ID
2664 MOV L$MCHN+$BFR,TEMP2 ; GET MAXIMUM CHANNELS
2665 ERRT$ DTMS2,MSGOUT ; DISPLAY INFO TO USER'S TERMINAL
2666 BR 10$ ; LOOK FOR NEXT BLOCK
2667 40$: BITB #F.FND,FLAGS ; WAS A DTE FOUND?
2668 BNE 60$ ; BR IF YES
2669 BIT #OP$DTE,$OPTON ; SHOW A SPECIFIC DTE?
2670 BNE 50$ ; BR IF YES
2671 ERRT$ NOINFO,MSGOUT ; DISPLAY NO INFORMATION MESSAGE
2672 BR 60$ ; AND EXIT
2673 50$: ERRT$ DTMS3,MSGOUT ; DTE NOT IN SYSTEM
2674 60$: RETURN
2675 ;

```

```

3153 .SBTTL DMX9S - SHOW MODULE X29-SERVER
3154
3155 ;+
3156 : DMX9S - SHOW MODULE X29-SERVER
3157 : INPUTS:
3158 : RGB - ADDRESS OF REQUEST BLOCK
3159 :
3160 : OUTPUTS:
3161 : INFORMATION DISPLAYED ON USER'S TERMINAL
3162 :
3163 :
3164 :
3165 :
3166 024226 016700 000000G DMX9S: MOV $PSIPT,RO ; GET POINTER TO PSI HOME BLOCK
3167 024232 062700 000040 ADD #H$X29C,RO ; POINT TO X29 DESCRIPTOR
3168 024236 GETRV RO,#2 ; READ IN DESCRIPTOR
3169 024254 005767 000000G TST $BFR ; IS X29 IN SYSTEM?
3170 024260 001004 BNE 10$ ; BR IF YES
3171 024262 ERRPT$ X9ER1,$EROUT ; ELSE DISPLAY ERROR MESSAGE
3172
3173 024272 032767 002200 000000G 10$: BIT #OP$DST!OP$KDS,$OPTON ; DESTINATION QUALIFIER SPECIFIED?
3174 024300 001405 BEQ 20$ ; BR IF NO
3175 024302 012700 000014 MOV #H$D29,RO ; POINT TO OFFSET INTO HOME BLOCK
3176 024306 CALL DMDST ; DISPLAY DESTINATION INFO
3177 024312 000426 BR 30$ ; AND EXIT
3178
3179 024314 20$: CEACC$ $BFR,RO ; CONVERT TO MAPPED ADDRESS
3180 024326 GETAD RO,#Q$LEN ; READ IN X29 DESCRIPTOR
3181 024344 016767 000000G 154462 MOV Q$CTIM+$BFR,TEMP1 ; GET COUNTER TIMER VALUE
3182 024352 016767 000002G 154502 MOV Q$MXVC+$BFR,TEMP2 ; GET MAXIMUM CIRCUITS
3183 024360 ERRPT$ X9MS1,MSGOUT ; DISPLAY COUNTER TIMER, MAX CIRCUITS
3184
3185 024370 30$: RETURN
3186
3187 : ERRORS
3188 :
3189 024372 ERMSG$ < X29 Descriptor block not in system>
3190 024440 ERBLK$ X9ER1
3191
3192 : MESSAGES
3193 :
3194 024444 ERMSG$ < Counter timer = %F, Maximum circuits = %F>
3195 024521 ERBLK$ X9MS1,<TEMP1,TEMP2>

```


3728
3729 026702
3730 026721
3731
3732

ERMSG\$ < State = %A%N>
ERBLK\$ PMSG3,<TEMP2>
.DSABL LC

```

4316 .SBITL DISPVC - DISPLAY PVC CIRCUIT INFORMATION ;[TD001]
4317 ;**=3 ;[TD001]
4318
4319
4320 *** DISPVC - DISPLAY PVC CIRCUIT INFORMATION ;[TD001]
4321 ;[TD001]
4322 INPUTS: ;[TD001]
4323 $BFR CONTAINS PVC NAME BLOCK ;[TD001]
4324 IF R4 = 0, PRINT HEADER INFORMATION ALONG WITH CIRCUIT INFORMATION ;[TD001]
4325 ELSE, PRINT ONLY CIRCUIT INFORMATION ;[TD001]
4326 ;[TD001]
4327 OUTPUTS: ;[TD001]
4328 CIRCUIT INFORMATION IS DISPLAYED ON USER'S TERMINAL ;[TD001]
4329 ;[TD001]
4330 ALL REGISTERS ARE PRESERVED ;[TD001]
4331 ;[TD001]
4332
4333 DISPVC: CALL $SAVAL ; SAVE ALL REGISTERS ;[TD001]
4334 TST R4 ; DISPLAY HEADER? ;[TD001]
4335 BNE 10$ ; BRANCH IF NO ;[TD001]
4336 ERRTT$ PVMMSG1,MSGOUT ; ELSE DISPLAY HEADER ;[TD001]
4337 ;[TD001]
4338
4339 DISPLAY CIRCUIT NAME ;[TD001]
4340 10$: MOV #TEMP3,R2 ; POINT TO TEMPORARY STORAGE ;[TD001]
4341 BIT #PF.XDF,$PFLAG ; CHECK FOR EXTENDED DATA FORMAT ;[TD001]
4342 BNE 12$ ; BRANCH IF EXTENDED ;[TD001]
4343 MOV #BFR+C$NAM,R0 ; GET ADDRESS OF CIRCUIT NAME ;[TD001]
4344 MOV #6,R1 ; LENGTH OF CIRCUIT NAME ;[TD001]
4345 BR 20$ ;[TD001]
4346 12$: MOV #BFR+C$GNAM,R0 ; GET ADDRESS OF EXTENDED CIRCUIT NAME ;[TD001]
4347 MOV $BFR+C$NML,R1 ; GET LENGTH OF CIRCUIT NAME ;[TD001]
4348 BEQ 22$ ; BRANCH IF NAME EMPTY ;[TD001]
4349 20$: MOVB (R0)+(R2)+ ; MOVE CIRCUIT NAME TO TEMP BUFFER ;[TD001]
4350 SOB R1,20$ ;[TD001]
4351 22$: CLRB (R2) ; TERMINATE STRING ;[TD001]
4352 ERRTT$ PVMMSG2,MSGOUT ; DISPLAY CIRCUIT NAME ;[TD001]
4353 ERRTT$ PVMMSG3,MSGOUT ; DISPLAY STATE AND TYPE ;[TD001]
4354 MOV C$LEN+$BFR,TEMP1 ; GET CHANNEL NUMBER ;[TD001]
4355 ERRTT$ PVMMSG4,MSGOUT ; DISPLAY CHANNEL NUMBER ;[TD001]
4356 RETURN ;[TD001]
4357 ;[TD001]
4358
4359 DISPLAY MESSAGES ;[TD001]
4360 ;[TD001]
4361
4362 ERMSG$ <%N% A AS OF %Y%N> ;[TD001]
4363 ERBLK$ PVMMSG1,<TEMP2,TEMP1> ;[TD001]
4364
4365 ERMSG$ <%NCIRCUIT = %A%N> ;[TD001]
4366 ERBLK$ PVMMSG2,<TEMP3> ;[TD001]
4367
4368 ERMSG$ < STATE = ON, TYPE = X25> ;[TD001]
4369 ERBLK$ PVMMSG3 ;[TD001]
4370
4371 ERMSG$ < CHANNEL = %F%N> ;[TD001]
4372 ERBLK$ PVMMSG4,<TEMP1> ;[TD001]

```

4792
4793
4794
4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829

035140 010046
035142 016701 000000G
035146 016702 000000G
035152 001407
035154 012100
035156 001403
035160 026016 000000G
035164 001403
035166 005302
035170 003371
035172 000261
035174 005226
035176 006016
035200 005741
035202 166701 000000G
035206 006316
035210

.SBTTL FNDPDV - FIND PDV ADDRESS

:+

FNDPDV - FIND PDV ADDRESS

INPUTS:

R0 = RAD50 PROCESS NAME

OUTPUTS:

CARRY CLEAR:

R0 = PDV ADDRESS OF PROCESS

R1 = PDV INDEX

CARRY SET:

PROCESS NOT FOUND

R2 DESTROYED

:-

```
FNDPDV: MOV    R0, -(SP)          ; SAVE NAME TO LOOK FOR
          MOV    $PDVTA, R1      ; GET PDV TABLE ADDRESS
          MOV    $PDVNM, R2      ; GET NUMBER OF PDV'S
          BEQ    30$             ; BR IF NONE
10$:      MOV    (R1)+, R0        ; GET PDV ADDRESS
          BEQ    20$             ; IF ZERO, SKIP IT
15$:      CMP    Z.NAM(R0), (SP) ; DOES THE NAME MATCH?
          BEQ    40$             ; BR IF YES (C-BIT CLEAR)
20$:      DEC    R2              ; DECREMENT COUNT
          BGT    10$            ; BR IF MORE TO CHECK
30$:      SEC                     ; INDICATE FAILURE
40$:      INC    (SP)+           ; CLEAN UP STACK
          ROR    (SP)            ; SAVE C-BIT
          TST    ~(R1)           ; CALCULATE PDV INDEX
          SUB    $PDVTA, R1
          ASL    (SP)            ; RESTORE CARRY
          RETURN
```

SYMBOL	VALUE	REFERENCES							
AALL	000135 R	#6-148 10-475							
ACMS1	016320 R	31-2342 #31-2376							
ACMS2A	016374 R	31-2358 #31-2379							
ACMS2B	016444 R	31-2361 #31-2382							
ACMS3	016520 R	31-2370 #31-2385							
ACNS1	017160 R	33-2449 #33-2493							
ACNS2	017202 R	33-2456 #33-2496							
ACNS3	017224 R	33-2464 #33-2499							
ACNS4	017252 R	33-2467 #33-2502							
ACNS5	017276 R	33-2475 #33-2505							
ACNS6	017346 R	33-2486 #33-2508							
ACNS7	017352 R	33-2477 #33-2511							
AERR1	003126 R	10-535 #10-548							
AERR2	003162 R	10-538 #10-551							
AKNOWN	000151 R	#6-149 10-521							
AMSG1	026322 R	49-3557 #49-3658							
AMSG2	026352 R	49-3568 #49-3661							
AMSG3	026424 R	49-3645 #49-3664							
AMSG4	026506 R	49-3648 #49-3667							
ARAMSK	= ***** GX	18-1049 18-1050	26-1462	26-1463	26-1586	26-1587	26-1634	26-1635	51-3770
ASPEC	000167 R	51-3771 10-491							
A.ACC	000020	#6-150 49-3605							
A.LNK	000000	#5-75							
A.NAM	000002	#5-75 10-500	49-3563						
A.PUD	000010	#5-75							
A.REM	000012	#5-75 49-3594							
A.UCB	000010	#5-75 10-508	10-528	49-3573	49-3577	49-3580			
BCDASC	035464 R	31-2347 35-2641	37-2771	40-3023	#64-4932	64-4947			
BF.TRC	= 000004	#5-88							
BITNUM	= 001204 R	#7-317 7-318	*19-1125	19-1134	19-1140	19-1151	19-1161	*19-1165	24-1367
BLDLIN	006500 R	24-1373 19-1095							
BLDMSK	006366 R	19-1112 #21-1218							
BLDNOD	006736 R	19-1099 #20-1189							
BROAD	000652 R	#6-219 53-3984	55-4417						
CACTIV	000520 R	#6-208 27-1937							
CHARAC	000017 R	#6-118 43-3223							
CHKBIT	007230 R	19-1136 #24-1366							
CHKSNK	005236 R	12-737 #17-977							
CIERR1	015156 R	27-2131 #27-2137							
CKNOWN	000540 R	#6-209 27-2079							
CKPREV	007174 R	19-1164 19-1168	#23-1344						
CKTYPE	024544 R	8-339 9-383	10-455	11-571	26-1434	27-1923	28-2155	29-2258	#43-3220
CKTYP1	024530 R	12-682 #43-3216							
CMSG1	032170 R	53-3952 #53-4284							
CMSG11	032572 R	#53-4309							
CMSG12	032600 R	53-4273 #53-4312							
CMSG2	032220 R	53-3960 #53-4287							
CMSG4	032274 R	53-4271 #53-4290							
CMSG5	032344 R	53-4224 #53-4293							
CMSG6	032406 R	53-4227 #53-4296							

VDIS	CREATED BY	MACRO	ON 15-JUL-85 AT 12:17	PAGE 14	I 9
SYMBOL CROSS REFERENCE			CREF	04.00	
SYMBOL	VALUE	REFERENCES			
R\$NWL	000104	33-2436			
R\$OWN	000010	31-2348	31-2352		
R\$PAL	000052	33-2465			
R\$USL	000031	33-2457			
R\$USR	000032	33-2459			
R\$SMPL	= *****	5-62	39-2869	39-2876	55-4466
R.ADD	000010	#5-94	12-681	22-1309	59-4721
		53-4214		25-1409	59-4729
R.CSR	000056	#5-88			59-4729
R.FLAG	000012	#5-94	26-1489	26-1558	26-1793
R.FLG	000064	#5-88			
R.LEN	000014	#5-94	22-1308	25-1408	26-1441
R.LNK	000000	#5-94			26-1513
R.LST	000104	#5-88	53-4050		53-4211
R.MAP	000110	#5-88			
R.NAM	000002	#5-94	18-1038	18-1039	18-1040
		51-3762	51-3763	51-3764	22-1314
R.QUE	000062	#5-88			26-1545
R.SLN	000070	#5-88			26-1547
R.SRV	000060	#5-88			53-4217
R.STA	000066	#5-88			
R.STBL	000210	#5-88			
SF.ACT	= 000200	#5-96			
SF.ENA	= 000100	#5-96	27-1974	53-4150	55-4511
SF.LPB	= 000004	#5-96	53-4192		
SF.MFL	= 000040	#5-96			
SF.PAC	= 000020	#5-96			
SF.REA	= 000010	#5-96			
SF.SER	= 000001	#5-96	53-4177		
SF.SVC	= 000002	#5-96			
SF.UNL	= 000040	#5-96			
SHUT	000103	#6-132			
SINGLE	000126	#6-143	48-3526		
SLASH	= 000057	#7-303	49-3606	49-3615	49-3628
SMSG1	002100	8-342	#8-353		49-3635
SMSG2	002242	8-343	#8-357		49-3637
SPACE	= 000040	#7-299	21-1237	21-1266	21-1267
		31-2339	34-2560	40-2943	22-1311
		53-4122	53-4221	55-4448	22-1316
		#6-117	43-3220		22-1325
SUMARY	000007	3-3-1	#8-363	9-385	10-457
SYNERR	002270	29-2260			11-573
		*10-460	*10-461	*10-462	12-684
SYSFDB	= *****	*22-1323	*25-1396	*25-1397	26-1436
		*26-1524	*26-1605	*26-1630	27-1925
		*27-1949	*27-2003	*27-2091	28-2157
		*34-2553	*34-2570	*35-2635	*18-1022
		*42-3180	*44-3280	*45-3364	*18-1023
		*53-4049	*53-4053	*53-4080	*18-1024
		*59-4733	*59-4743	*59-4749	*22-1297
		#5-96	53-3973		*22-1298
S.COST	000001	#5-88			*22-1301
S.CTL	000006				*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26-1484
					*26

FILEID**VENA

```

VV      VV  EEEEEEEEE  NN      NN      AAAAAA
VV      VV  EEEEEEEEE  NN      NN      AAAAAA
VV      VV  EE          NN      NN      AA      AA
VV      VV  EE          NN      NN      AA      AA
VV      VV  EE          NNNN     NN      AA      AA
VV      VV  EE          NNNN     NN      AA      AA
VV      VV  EEEEEEEEE  NN      NN      AA      AA
VV      VV  EEEEEEEEE  NN      NN      AA      AA
VV      VV  EE          NN      NNNN   AAAAAAAAAA
VV      VV  EE          NN      NNNN   AAAAAAAAAA
VV      VV  EE          NN      NN      AA      AA
VV      VV  EE          NN      NN      AA      AA
VV      VV  EEEEEEEEE  NN      NN      AA      AA
VV      VV  EEEEEEEEE  NN      NN      AA      AA

```

```

....
....
....
....

```

```

LL      SSSSSSSS  TTTTTTTTTT
LL      SSSSSSSS  TTTTTTTTTT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LL      SSSSSS    TT
LL      SSSSSS    TT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LL      SS        TT
LLLLLLLLLL  SSSSSSSS  TT
LLLLLLLLLL  SSSSSSSS  TT

```

```

378 .SBTTL $VENA - ENABLE/DISABLE LINE
379
380 *** - $VENA - ENABLE/DISABLE LINE
381
382 INPUTS:
383 $SLTA - SYSTEM LINE TABLE ADDRESS
384 $SLN - SYSTEM LINE NUMBER (NON - DLX ONLY SYSTEMS)
385 $TRIB - TRIBUTARY NUMBER
386
387 OUTPUTS:
388 XPT DATA BASE UPDATED IF NON- DLX ONLY SYSTEM
389 C-BIT SUCCESS/FAILURE
390
391
392
393 000242 052767 000001 000126' $VSR:: BIS #SER,SERFLG ; SET CIRCUIT SERVICE FLAG
394 000250 $VENA::
395 .MCALL VNPBPT
396 000250 VNPBPT VENA
397 000250 CALL $SAVAL ; SAVE REGISTERS
398
399 000254 105067 000125' CLRBL DLXPX ; ASSUME DLX NOT IN SYSTEM
400 000260 012700 015370 MOV #*RDLX,R0 ; GET DLX PROCESS NAME IN RAD50
401 000264 CALL $FPDV ; LOOK FOR DLX PDV
402 000270 103402 BCS 10$ ; BR IF DLX NOT IN SYSTEM
403 000272 110167 000125' MOVBL R1,DLXPX ; SAVE DLX PDV INDEX
404
405 000276 105067 000124' 10$ CLRBL SNAPX ; ASSUME SNA NOT IN SYSTEM
406 000302 012700 074361 MOV #*RSNA,R0 ; GET PROCESS NAME
407 000306 CALL $FPDV ; LOOK FOR SNA PDV
408 000312 103402 BCS 20$ ; BR IF SNA NOT IN SYSTEM
409 000314 110167 000124' MOVBL R1,SAPX ; SAVE SNA PDV INDEX
410
411 000320 005005 20$ CLR R5 ; ZERO => "DON'T REWRITE DESCRIPTOR"
412 000322 CALL CHKXPT ; IS XPT CONTEXT CORRECT ?
413 000326 103004 BCC 30$ ; IF CS, YES
414 000330 052767 000000G 000000G BIS #VF,DLX,$VFLAG ; SET TEMP VFLAG BIT FOR DLX ONLY SYSTEM
415 000336 000466 BR 60$ ; SKIP XPT STUFF
416 000340 016704 000000G 30$ MOV $SLN,R4 ; GET SYSTEM LINE NUMBER
417 000344 006304 ASL R4 ; FORM WORD INDEX
418 000346 066704 000000G ADD $LLCTA,R4 ; INDEX INTO REVERSE MAPPING TABLE
419 000352 032714 100000 BIT #100000,(R4) ; MULTIPOINT LINE ?
420 000356 001404 BEQ 40$ ; IF EQ, NO -- OKAY
421 000360 011404 MOV (R4),R4 ; GET ADDRESS/2 OF STATION TABLE
422 000362 066704 000000G ADD $TRIB,R4 ; POINT TO STATION IN TABLE
423 000366 006304 ASL R4 ; CONVERT TO REAL ADDRESS
424 000370 126401 000001 40$ CMPB 1(R4),R1 ; IS THIS SYSTEM LINE ASSIGNED TO XPT ?
425 000374 001411 BEQ 50$ ; BR IF YES
426 000376 126467 000001 000124' CMPB 1(R4),SNAPX ; IS THIS LINE ASSIGNED TO SNA ?
427 000404 001443 BEQ 60$ ; BR IF YES
428 000406 126467 000001 000125' CMPB 1(R4),DLXPX ; IS THIS LINE OWNED BY DLX?
429 000414 001172 BNE 190$ ; BR IF NO -- ERROR
430 000416 000436 BR 60$ ; GO SET LINE ON
431 000420 111467 000070' 50$ MOVBL (R4),CHAN ; STORE CHANNEL NUMBER
432 000424 062700 000000G ADD #2,MAP,R0 ; FORM ADDRESS OF XPT CHANNEL TABLE
433 000430 016746 000070' MOV CHAN,-(SP) ; TURN CHANNEL
434 000434 006316 ASL (SP) ; NUMBER INTO

```

937
 938
 939
 940
 941
 942

002504 005726
 002506
 002516

; ERROR CONDITIONS

101\$: TST (SP)+
 ERRPT\$ ERR35,\$EROUT
 102\$: ERRPT\$ ERR21,\$EROUT
 .DSABL LSB

; CLEAN UP STACK
 ; LINE IN WRONG STATE
 ; LINE NOT IN SYSTEM


```

1528 005030 103401      BCS 25$      ; IF CS, NO .. OKAY
1529 005032 010002      MOV R0,R2      ; ELSE, SAVE NODE BLOCK ADDRESS
1530 005034 016701 000012G 25$: MOV R0B+LR.ADD,R1 ; GET THE NODE ADDRESS
1531 005040      CALL $ENADD      ; TRY TO FIND THE REMOTE BLOCK
1532 005044 103016      BCC 27$      ; IF CC, FOUND IT
1533 005046 005702      TST R2      ; WAS THERE A MATCH ON THE NODE NAME ?
1534 005050 001421      BEQ 28$      ; IF EQ, NO .. CREATE NEW BLOCK
1535 005052 012701 000044G      MOV #R0B+LO.NAM,R1 ; ELSE, GET NODE NAME STRING
1536 005056      CALL $FNAM      ; AND FIND THAT BLOCK AGAIN
1537 005062 103001      BCC 26$      ; IF CC, OKAY
1538 005064 000004      JOT      ; ELSE, I DUNNO
1539 005066 016767 000012G 000010G 26$: MOV R0B+LR.ADD,$BFR+R.ADD ; AND STORE NODE ADDRESS
1540 005074      PUTAD      ; RE-WRITE THE NODE BLOCK
1541 005100 000477      BR 40$      ; ELSE, NO CHANGE
1542
1543      ; MATCH ON NODE ADDRESS
1544
1545 005102 005702      27$: TST R2      ; WAS THERE A MATCH ON THE NODE NAME ?
1546 005104 001726      BEQ 10$      ; IF EQ, NO .. OKAY
1547 005106 020002      CMP R0,R2      ; ELSE, BLOCKS MUST BE THE SAME
1548 005110 001126      BNE 104$      ; IF NE, NO .. ERROR
1549 005112 000472      BR 40$      ; ELSE, LEAVE
1550
1551      ; CREATE NEW REMOTE NODE ENTRY
1552
1553 005114 012701 000014      28$: MOV #R.LEN,R1      ; GET LENGTH OF REMOTE BLOCK
1554 005120      CALL $ALPOL      ; TRY TO ALLOCATE A REMOTE BLOCK
1555 005124 103510      BCS 102$      ; IF CS, ALLOCATION FAILURE
1556 005126 010067 000164'      MOV R0,CURUM      ; SAVE UNMAPPED ADDRESS
1557 005132 012701 000000G      MOV #BFR,R1      ; ELSE, POINT TO I/O BUFFER
1558 005136 062701 000002      ADD #R.NAM,R1      ; POINT TO NAME FIELD
1559 005142 012702 000044G      MOV #R0B+LO.NAM,R2 ; POINT TO INPUT NAME
1560 005146 012221      MOV (R2)+,(R1)+      ; STORE NAME IN REMOTE BLOCK
1561 005150 012221      MOV (R2)+,(R1)+      ; ...
1562 005152 012221      MOV (R2)+,(R1)+      ; ...
1563 005154 162701 000010      SUB #R.NAM+6,R1      ; POINT TO START OF BLOCK AGAIN
1564 005160 016761 000012G 000010      MOV R0B+LR.ADD,R1.ADD(R1) ; STORE NODE ADDRESS
1565 005166 005061 000012      CLR R.FLAG(R1)      ; INITIALIZE FLAGS WORD
1566
1567      ; INSERT ENTRY INTO REMOTE LIST
1568
1569 005172      REMINS: CEACCS$ R0,R1      ; CONVERT TO MAPPED ADDRESS
1570 005202      PUTAD R1,#R.LEN      ; WRITE OUT REMOTE
1571
1572 005220      SWTCH $HMBUF      ; USE TEMP BUFFER
1573 005226      GETRY $DECP,#D$END      ; READ DECNET HOME BLOCK
1574 005246      SWTCH      ; RESET DEFAULT BUFFER
1575
1576 005254 016701 000010G      MOV $BFR+R.ADD,R1      ; GET SEARCH KEY
1577 005260 012702 000004G      MOV $HMBUF+D$RNN+2,R2 ; POINT TO LISTHEAD
1578 005264      CALL FINREM      ; FIND PLACE FOR REMOTE BLOCK
1579 005270 012702 000004G      35$: MOV $HMBUF+D$RNN+2,R2 ; POINT TO LISTHEAD
1580 005274      CALL QINS      ; AND INSERT ENTRY INTO LIST
1581 005300      40$:
1582 005300      SWTCHO $HMBUF      ; USE TEMP BUFFER
1583 005306      PUTRC $DECP,#D$END      ; WRITE DECNET HOME BLOCK
1584 005326      SWTCHO      ; RESET TEMP BUFFER

```

```

1985 006670 001002      BNE      8$      ; IF YES - BRANCH
1986 006672      CALLR    30$      ; IF EQ, NO - MUST BE DISMOUNT
1987 006676      8$:      CALL    $ALL15  ; GET I/O PACKET
1988 006702      BCC      16$      ; BR IF SUCCESS
1989 006704      JMP      109$     ; ELSE NODE POOL EMPTY
1990 006710      GETRV    R5,TLGTH  ; REREAD TCB
1991 006726      TST      T.RCVL+$BFR ; ANY PACKETS QUEUED ?
1992 006732      BEQ      17$      ; BR IF NO
1993 006734      GETRV    T.RCVL+$BFR,#36 ; READ IN PACKET
1994 006754      032767 010000 000006G BIT    #LS.CXO,LR.STS+6+$BFR ; IS SET EXE STATE ON ALREADY QUEUED?
1995 006762      001402      BEQ      161$    ; BR IF NO
1996 006764      000167 000476      JMP      110$    ; ELSE NOTHING TO DO
1997 006770      052767 010000 000006G 161$: BIS    #LS.CXO,LR.STS+6+$BFR ; ELSE SET SET EXE STATE ON ALSO
1998 006776      PUTRC    $DEA16    ; REWRITE PACKET
1999 007002      CALL      25$      ; DEALLOCATE PACKET
2000 007006      000461      BR      17$      ; AND CONTINUE
2001 007010      010067 000012G      MOV      R0,T.RCVL+$BFR ; POINT LISHEAD AT FIRST PACKET
2002 007014      010067 000014G      MOV      R0,T.RCVL+2+$BFR ;
2003 007020      PUTRC    ; REWRITE NTINIT TCB ADDRESS
2004
2005 007024      005067 000000G      CLR      $BFR      ; CLEAR LINK WORD IN PACKET
2006 007030      012767 131574 000002G      MOV      #*R...,$BFR+2 ; LOAD RAD50 NAME
2007 007036      012767 105700 000004G      MOV      #*RVNP,$BFR+4 ; *RVNP
2008 007044      012767 010001 000006G      MOV      #LS.CEX!LS.CXO,LR.STS+6+$BFR ; ' SET STA LOC ON' -> 'SET CEX'
2009 007052      042767 020000 000006G      BIC      #LS.UNF,LR.STS+6+$BFR ; ASSUME NTINIT IS TO REMAIN FIXED
2010 007060      032767 000000G 000000G      BIT      #VF.FIX,$VFLAG ; IS NTINIT TO REMAIN FIXED ?
2011 007066      001003      BNE      20$      ; YES
2012 007070      052767 020000 000006G      BIS      #LS.UNF,LR.STS+6+$BFR ; SET FLAG THAT UNFIXES NTINIT
2013 007076      016767 000000G 000040G      MDV      ..COO,$BFR+32. ; ERRORS GO TO COO:
2014 007104      012767 000401 000042G      MOV      #401,$BFR+34. ; [1,1]
2015 007112      PUTRC    R0,R1 ; REWRITE NODE TO POOL
2016 007126      012704 000001      MOV      #TICKS,R4 ; NUMBER OF TICKS
2017 007132      CALL      RUN ; RUN NTINIT
2018 007136      103002      BCC      24$      ; IF SUCCESS - BRANCH
2019 007140      000167 000206      JMP      104$    ; NO NODES IN POOL
2020 007144      24$:      SWITCHO ; POINT BACK AT DEFAULT BUFFER
2021
2022 007152      25$:      CALL      ALPLB ; ALLOCATE PLB DATA BASE
2023 007156      103465      BCS      46$      ; IF CS, FAILED TO ALLOCATE PLB'S
2024 007160      052767 000000G 000000G      BIS      #VF.ACP,$VFLAG ; INDICATE SUCCESS
2025
2026
2027 ;
2028 ; STORE BIAS OF AVAILABLE TEMPORARY STORAGE IN RDB FOR TERBOT AND
2029 ; NTINIT TO USE FOR MANIPULATING HOST ADDRESS (11S ONLY)
2030
2031 007166      032767 000000G 000000G 45$: BIT      #VF.SS,$VFLAG ; IS THIS AN 11S SYSTEM?
2032 007174      001461      BEQ      100$     ; BR IF NO
2033 007176      016767 000000G 000000G      MOV      $RDBLH,$BFR ; GET BIAS FOR RDB'S
2034 007204      016767 000002G 000002G      MOV      $RDBLH+2,$BFR+2 ; GET VIRTUAL OFFSET
2035 007212      022767 140070 000002G 47$: CMP      #140070,$BFR+2 ; CHECK FOR AVAILABLE SPACE
2036 007220      002015      BGE      48$      ; IF O.K. - BRANCH
2037 007222      016704 000002G      MOV      $BFR+2,R4 ; GET VIRTUAL OFFSET
2038 007226      016767 000000G 000000G      MOV      $BFR,$RBLCK ; GET APR BIAS
2039
2040 007234      GETAD      R4,#4 ; READ 4 POINTER BYTES OF NEXT RDB
2041 007252      000757      BR      47$      ; CONTINUE

```

```

2595 012312 042767 177700 000000G      BIC      #^C<FF.MSK>,$RSIZE ; ...
2596
2597 012320 022767 000016 000000G      CMP      #F.LEN,$RSIZE ; IS IT THE CORRECT LENGTH
2598 012326 001402                      BEQ      5$ ; IF YES - BRANCH
2599 012330                      GETAD    ; IF NO - READ ENTIRE FILTER BLOCK
2600
2601 012334 032767 000010 000014G 5$:    BIT      #FF.LIN,F.FLG+$BFR ; IS THIS BLOCK SPECIFIED BY LINE ?
2602 012342 001741                      BEQ      1$ ; IF NO BRANCH
2603 012344 116746 000034G      MOVB     LD,TRB+R0B,-(SP) ; COPY TRIB NUMBER
2604 012350 000316                      SWAB     (SP) ; FLIP BYTES
2605 012352 116716 000000G      MOVB     $SLN,(SP) ; COPY SYSTEM LINE NUMBER
2606 012356 026726 000020G      CMP      F.LIN+$BFR,(SP)+ ; MATCH ON LINE ?
2607 012362 001331                      BNE      1$
2608
2609 012364 012702 000004G      MOV      #F.EVT+$BFR,R2 ; GET ADDRESS OF EVENT MASK
2610 012370 012703 000030G      MOV      #F.CEV+$BFR,R3 ; GET CLEAR EVENT MASK ADDRESS
2611
2612 012374 012746 000004          10$:   MOV      #4,-(SP) ; MOVE FOUR WORDS
2613 012400 011213                      MOV      (R2),(R3) ; SET UP CLEAR MASK
2614 012402 005123                      COM      (R3)+ ; COMPLEMENT MASK
2615 012404 005022                      CLR      (R2)+ ; CLE KNO EVE
2616 012406 005316                      DEC      (SP) ; COUNT ONE MORE MASK SETUP
2617 012410 001373                      BNE     10$ ; IF MORE - BRANCH
2618 012412 005726                      TST      (SP)+ ; CLEAN STACK
2619
2620 012414                      PUTAD    ; UPDATE FILTER BLOCK
2621 012420 000712                      BR      1$ ; CONTINUE
2622

```

CHFIL - CHANGE EVENT FILTERS

.SBTTL CHFIL - CHANGE EVENT FILTERS

3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153

```

+
: CHFIL - CHANGE EVENT FILTERS
: INPUTS:
:   RQB - REQUEST BLOCK FOR SET/CLEAR LOGGING COMMAND
: OUTPUTS:
:   THE EVENT FILTERS ARE UPDATED
:   R5 IS PRESERVED
-

CHFIL:  MOV    R5, -(SP)           ; SAVE R5
        CALL   STQUAL            ; SET UP QUALIFIER FLAGS WORD
        BIT    #QF$KNO,$QUAL     ; IS THIS FOR KNOWN LOGGING?
        BEQ    30$               ; BR IF NO
        MOV    #SNKTYP,R5        ; POINT TO SINK TYPE TABLE
        CLR    R3                ; NO BITS TO CLEAR
        BIC    R3,EVQUAL         ; CLEAR BIT FOR PROCESSED SINK TYPE
        MOV    (R5)+,R3          ; GET NEXT SINK TYPE BIT
        BEQ    70$               ; BR IF END OF TABLE
        BIS    R3,EVQUAL         ; SET SINK TYPE FLAG TO OPERATE ON
        BIT    #OP$KEV,$OPTON    ; IS THIS FOR KNOWN EVENTS?
        BEQ    50$               ; BR IF NO
        MOV    #EVTVAL,R4        ; POINT TO EVENT VALIDITY TABLE
        MOV    (R4)+,R2          ; STORE EVENT CLASS
        BLT    60$               ; BR IF END OF TABLE
        ASL    6,R2              ; MOVE EVENT CLASS TO BITS 6-15
        MOV    R2,RQB+LO.CLS     ; STORE EVENT CLASS
        MOV    #RQB+LO.EVT,R1    ; POINT TO EVENT MASKS
        .REPT 4
        MOV    (R4)+,(R1)        ; STORE EVENT MASKS
        COM    (R1)+             ; TABLE IS COMPLEMENT OF VALID MASKS
        .ENDR

        MOV    RQB+LO.CLS,CLASS  ; SAVE EVENT CLASS
        CALL   FILPRO            ; PROCESS FILTER MASKS
        BIT    #OP$KEV,$OPTON    ; IS THIS FOR KNOWN EVENTS?
        BNE    40$               ; BR IF YES
        BIT    #QF$KNO,$QUAL     ; IS THIS FOR KNOWN LOGGING?
        BNE    20$               ; BR IF YES
        MOV    (SP)+,R5          ; RESTORE R5
        RETURN

```

```

1416 .SBTTL $DNOD - SHOW NODE
1417
1418 ;+
1419 ; $DNOD - SHOW KNOWN NODES
1420 ; SHOW LOOP NODES
1421 ; SHOW NODE X
1422 ; SHOW EXECUTOR
1423
1424 ; INPUTS: NONE
1425
1426 ; OUTPUTS: THE NODE INFORMATION IS DISPLAYED ON THE USER'S TERMINAL
1427
1428 ; ALL REGISTERS ARE DESTROYED
1429
1430 ; -
1431
1432 $DNOD::
1433     007426          CALL    CKTYPE          ; GET COMMAND TYPE
1434     007426          BCC     10$             ; BR IF SUCCESS
1435     007432          103004          ERRPT$ SYNERR,$EROUT ; ELSE SYNTAX ERROR
1436     007434          012701 001062' 10$: MOV    #TEMP2,R1 ; POINT TO STORAGE FOR HEADER
1437     007444
1438
1439     007450          GETRV   $DECPY,$D$END    ; READ DECNET HOME BLOCK
1440     007470          012703 000004G MOV    #$BFR+$SRNN+2,R3 ; STORE REMOTE LISTHEAD
1441     007474          012705 000014 MOV    #R.LEN,R5 ; GET LENGTH OF REMOTE NODE BLOCK
1442     007500          005767 000000G TST     $QUAL ; SHOW SPECIFIC NODE?
1443     007504          001535 BEQ     130$ ; BR IF YES
1444     007506          005713 TST     (R3) ; ANY NODES IN SYSTEM?
1445     007510          001002 B""      20$ ; BR IF YES
1446     007512          000167 002306 JMP     520$ ; ELSE FINISHED
1447     007516          022767 000010 000000G 20$: CMP    #OF$LOO,$QUAL ; SHOW LOOP NODES?
1448     007524          001002 BNE     30$ ; BR IF NO
1449     007526          000167 002146 JMP     450$ ; ELSE SHOW LOOP NODES
1450
1451 ; SHOW KNOWN NODES
1452
1453     007532          012700 000362' 30$: MOV    #NKNOWN,R0 ; MUST BE SHOW KNOWN NODES
1454     007536          112021 40$: MOVB   (R0)+,(R1)+ ; STORE THE STRING
1455     007540          001376 BNE     40$ ;
1456     007542          ERRPT$ NMSG1,MSGOUT ; DISPLAY THE HEADER
1457
1458 ; DISPLAY EXECUTOR ADDRESS AND NAME
1459
1460     007552          CALL    GTXEC          ; GET EXECUTOR ADDRESS AND NAME
1461     007556          016701 171252 MOV    TEMP1,R1 ; COPY EXECUTOR ADDRESS
1462     007562          042767 000000G 171244 BIC    #ARAMSK,TEMP1 ; STRIP AREA BITS
1463     007570          042701 000000C BIC    #*CARAMSK,R1 ; GET AREA NUMBER
1464
1465     000012          .REPT    10. ;
1466     .ASR            R1 ; CONVERT TO AREA
1467     .ENDM
1468     007620          010167 171732 MOV    R1,TEMP7 ; STORE AREA FOR OUTPUT
1469
1470     007624          ERRPT$ NMSG2,MSGOUT ; PRINT EXECUTOR NODE ADDRESS
1471
1472 ;

```

VDIS - VNP SHOW COMMANDS
\$DCIR - SHOW CIRCUIT

MACRO V05.03b Monday 15-Jul-85 12:14^{J 2} Page 27-4

2135
2136 015126
2137 015154

;

ERMSG\$ < CIRCUIT NOT IN SYSTEM>
ERBLK\$ CIERR1

VDIS - VNP SHOW COMMANDS
\$DCIR - SHOW CIRCUIT

MACRO V05.03b Monday 15-Jul-85 12:14^{K 2} Page 28

```
2676                                     :  MESSAGES
2677                                     :
2678 020766                             ERMSG$ < DTE = %A>
2679 020777                             ERBLK$ DTMS1,<TEMP1>
2680
2681 021004                             ERMSG$ <      State = %A, Maximum channels = %D, Line = %A%N>
2682 021066                             ERBLK$ DTMS2,<TEMP1,TEMP2,TEMP3>
2683
2684 021100                             ERMSG$ < DTE not in system>
2685 021122                             ERBLK$ DTMS3
```

```

3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216 024530 012700 000000' CKTYP1: .ENABL LSB
3217 024534 022767 001000 000000G CMP #EVENT,RO ; ASSUME EVENTS TO BE DISPLAYED
3218 024542 001415 BEQ 10$ #TY$EVE,$TYPE ; DISPLAY EVENTS?
3219 ; BR IF YES
3220 024544 012700 000007' CKTYPE: MOV #SUMARY,RO ; ASSUME SUMMARY TO BE DISPLAYED
3221 024550 005767 000000G TST $TYPE ; DISPLAY SUMMARY?
3222 024554 001410 BEQ 10$ ; BR IF YES
3223 024556 012700 000017' MOV #CHARAC,RO ; ASSUME CHARACTERISTICS TO BE DISPLAYED
3224 024562 022767 000400 000000G CMP #TY$CHA,$TYPE ; DISPLAY CHARACTERISTICS?
3225 024570 001402 BEQ 10$ ; BR IF YES
3226 024572 000261 SEC ; ELSE SYNTAX ERROR
3227 024574 000410 BR 30$ ; AND EXIT
3228 024576 10$: DIR$ #TIATT ; ATTACH TERMINAL
3229 024604 012701 001034' MOV #TEMP1,R1 ; GET ADDRESS OF STRING TO BE DISPLAYED
3230 024610 112021 20$: MOVB (R0)+,(R1)+ ; STORE IT
3231 024612 001376 BNE 20$
3232 024614 000241 CLC
3233 024616 30$: RETURN
3234 .DSABL LSB

```



```

3734 .SBTTL DISNOD - DISPLAY NODE INFORMATION
3735
3736 ;+
3737 DISNOD - DISPLAY NODE INFORMATION
3738
3739 INPUTS:
3740 R3 = POINTER TO REMOTE NODE BLOCK
3741 R4 = 0 IF HEADER IS TO BE PRINTED ALONG WITH NODE INFORMATION
3742 ELSE, ONLY NODE INFORMATION IS PRINTED
3743
3744 OUTPUTS:
3745 NODE INFORMATION IS DISPLAYED ON USER'S TERMINAL
3746
3747 ALL REGISTERS ARE PRESERVED
3748
3749
3750
3751
3752 DISNOD: SAVRG <R1,R2> ; SAVE R1
3753 TST R4 ; DISPLAY HEADER?
3754 BNE 10$ ; BR IF NO
3755 MOV #TEMP1,R1 ; STORAGE FOR COLUMN HEADER STRING
3756 ERRPT$ DNMSG1,MSGOUT ; DISPLAY COLUMN HEADER
3757
3758 ; DISPLAY NODE INFORMATION
3759
3760 026752 016367 000010 152222 10$: MOV R,ADD(R3),TEMP3 ; SAVE NODE ADDRESS
3761 026760 012701 001034' MOV #TEMP1,R1 ; STORAGE FOR NODE NAME
3762 026764 016321 000002 MOV R,NAM(R3),(R1)+ ; STORE THE NAME
3763 026770 016321 000004 MOV R,NAM+2(R3),(R1)+ ; ...
3764 026774 016321 000006 MOV R,NAM+4(R3),(R1)+ ; ...
3765 027000 122741 000040 20$: CMPB #SPACE,-(R1) ; STRIP OFF TRAILING BLANKS
3766 027004 001775 20$ BEQ 20$ ; ...
3767 027006 112761 000000 000001 MOVB #0,1(R1) ; CREATE ASCIZ STRING
3768
3769 027014 016367 000010 152532 MOV R,ADD(R3),TEMP6 ; GET NODE ADDRESS
3770 027022 042767 000000G 152152 BIC #ARAMSK,TEMP3 ; MASK OUT AREA BITS
3771 027030 042767 000000C 152516 BIC #*CARAMSK,TEMP6 ; MASK OUT NODE ADDRESS BITS
3772 027036 000241 CLC ; CLEAR C-BIT FOR DIVIDE
3773
3774 000012 .REPT 10.
3775 ROR TEMP6 ; DIVIDE BY 1024.
3776 .ENDM
3777
3778 027110 022767 000012 152436 30$: CMP #10.,TEMP6 ; TWO DIGITS ?
3779 027116 003440 BLE 70$ ; IF YES - BRANCH
3780
3781 027120 026727 152056 000012 CMP TEMP3,#10. ; IS ADDRESS ONE DIGIT
3782 027126 002005 BGE 40$ ; IF NO - BRANCH
3783 027130 ERRPT$ DNMSG8,MSGOUT ; PRINT AREA ADDRESS
3784 027140 000466 BR 110$ ; CONTINUE
3785
3786 027142 026727 152034 000144 40$: CMP TEMP3,#100. ; IS ADDRESS TWO DIGITS
3787 027150 002005 BGE 50$ ; IF NO - BRANCH
3788 027152 ERRPT$ DNMSG9,MSGOUT ; PRINT AREA ADDRESS
3789 027162 000455 BR 110$ ; CONTINUE
3790

```

VDIS - VNP SHOW COMMANDS MACRO V05.03b Monday 15-Jul-85 12:14^{J 6} Page 54-1
DISPVC - DISPLAY PVC CIRCUIT INFORMATION

4373

;[TD001]

```

4831                                     .SBTTL FNDUCB - GET UCB OF NS: DEVICE
4832
4833                                     ;+
4834                                     : FNDUCB - GET UCB OF NS: DEVICE
4835                                     :
4836                                     : INPUTS:
4837                                     : NONE
4838                                     :
4839                                     : OUTPUTS:
4840                                     : CARRY CLEAR - $BFR CONTAINS UCB OF NS: DEVICE
4841                                     :
4842                                     : CARRY SET - NS: DEVICE NOT IN SYSTEM
4843                                     :
4844                                     : -
4845
4846
4847 035212 016767 000000G 000000G FNDUCB: MOV    $DEVHD,$BFR      ; GET DEVICE TABLE LISTHEAD
4848 035220 016767 000000G 000000G 10$: MOV    $BFR,$RADDR      ; GET NEXT DCB IN LIST
4849 035226 001422                                     BEQ    30$              ; BR IF NOT END OF LIST
4850 035230                                     20$: GETRV   #D.LEN      ; READ IN DCB
4851 035242 026727 000004G 051516      CMP    $BFR+D.NAM,#'NS    ; IS THIS THE NS: DEVICE?
4852 035250 001363      BNE    10$              ; BR IF NO
4853 035252      GETRV   $BFR+D.UCB,#U.LEN    ; READ IN THE NS: UCB
4854 035272 000401      BR    40$              ; FINISHED
4855 035274 000261      30$: SEC                ; NS: DEVICE NOT IN SYSTEM
4856 035276      40$: RETURN

```

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 2 J 8
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
CMSG7	032432 R	53-4127 #53-4299
CMSG8	032506 R	53-4093 #53-4302
CMSG9	032546 R	53-4095 #53-4305
COMMA	= 000054	#7-306 19-1150
CSPEC	000557 R	#6-210 27-1991
CUNDEC	035300 R	37-2756 #63-4871
CVTDEV	005214 R	15-895 15-899 #16-950
CVTHEX	001575 R	#6-259 6-260 65-4976 65-4979
C\$GNAM	000016	27-2008 54-4346
C\$LCN	000010	24-4354
C\$LEN	000016	27-1951 27-2007 27-2011 27-2093
C\$NAM	000002	27-2012 54-4343
C\$NML	000002	27-2009 54-4347
DASH	= 000055	#7-301 19-1158 19-1160 21-1240 21-1249 56-4583 56-4593 57-4635 57-4645
DEFAULT	000110 R	#6-137 48-3496
DF.CRC	= 000010	#5-88
DF.FOC	= 000020	#5-88
DF.RAN	= 000040	#5-88
DF.UNK	= 000100	#5-88
DISABL	000507 R	#6-203 53-4179
DISALI	025666 R	10-481 10-510 10-530 #49-3555
DISCLR	030042 R	27-1976 27-1985 27-2064 27-2073 27-2112 27-2121 #53-3949
D:SLIN	033074 R	28-2180 28-2210 28-2224 #55-4395
DISLOO	027630 R	26-1511 26-1560 26-1798 #52-3875
DISNOD	026726 R	26-1491 26-1562 #51-3752
DISOBJ	025144 R	9-402 9-419 #48-3452
DISPRO	026516 R	11-593 11-609 11-621 #50-3693
DISPVC	032602 R	27-1957 27-2016 27-2099 #54-4333
DLOEVT	005700 R	12-742 #19-1088
DLONAM	004756 R	12-718 #15-882
DLOSNK	005360 R	17-991 17-996 #18-1015
DLOSTA	004716 R	12-714 #14-857
DLOTYP	004600 R	12-709 #13-820
DMDST	022342 R	39-2857 #40-2925 42-3176
DMDE	020520 R	34-2534 #35-2633
DMGRP	021226 R	34-2539 #37-2739
DMXAC	015724 R	29-2286 #30-2305
DMXACD	015760 R	30-2310 #31-2326
DMXACN	016562 R	30-2313 #33-2425
DMXPR	017354 R	29-2287 #34-2526
DMX55	021762 R	29-2288 #39-2849
DMX95	024226 R	29-2289 #42-3166
DNMSG1	027352 R	51-3756 #51-3825
DNMSG2	027366 R	#51-3828
DNMSG3	027406 R	51-3811 #51-3831
DNMSG4	027432 R	51-3793 #51-3834
DNMSG5	027456 R	51-3806 #51-3837
DNMSG6	027502 R	51-3801 #51-3840
DNMSG7	027524 R	51-3814 #51-3843
DNMSG8	027552 R	51-3783 #51-3846
DNMSG9	027576 R	51-3788 #51-3849
DNMS10	027620 R	51-3796 #51-3852

SYMBOL CROSS REFERENCE

CREF 04.00

SYMBOL VALUE REFERENCES

S.DLCF	000102	53-4025							
S.ERR	000010	#5-88							
S.FLG	000000	#5-96	27-1974	53-4150	53-4177	53-4192	55-4511		
S.LBE	000044	#5-88							
S.LEN	000004	#5-96	27-1978	27-2051	27-2067	27-2115			
S.LGTH	= 000050	#5-88	53-4053						
S.LIN	000004	#5-88							
S.LINK	177776	#5-88	5-88						
S.LSA	000005	#5-88	53-4056						
S.NMST	000002	#5-96	53-4112						
S.OWNR	000003	#5-96	53-4245						
S.PSA	000003	#5-88	53-4060						
S.RBE	000042	#5-88							
S.RCB	000020	#5-88							
S.RCV	000014	#5-88							
S.RDE	000034	#5-88							
S.RTY	000001	#5-88							
S.STA	000000	#5-88							
S.STLN	000035	53-4027							
S.STPN	000034	53-4029							
S.TLZ	000012	#5-88							
S.TMO	000040	#5-88							
S.XDE	000036	#5-88							
S.XID	000002	#5-88							
S.XMB	000030	#5-88							
S.XMP	000024	#5-88							
TEMP1	001034 R	#6-249	7-314	8-353	12-761	12-764	12-767	12-770	12-773
		12-786	13-836	14-857	18-1037	*18-1048	*18-1050	25-1400	*25-1401
		25-1409	26-1461	*26-1462	26-1585	*26-1586	*26-1625	*26-1631	*26-1665
		*26-1667	*26-1690	26-1831	26-1834	26-1837	26-1840	26-1846	26-1849
		26-1857	26-1878	26-1881	26-1897	29-2281	31-2335	31-2346	31-2376
		31-2382	33-2443	33-2451	33-2460	33-2471	33-2493	33-2496	33-2499
		34-2554	34-2558	*34-2576	*34-2583	*34-2584	*34-2591	*34-2592	34-2608
		34-2614	34-2617	35-2640	35-2648	35-2679	35-2682	37-2745	37-2748
		37-2788	37-2791	*39-2895	39-2907	40-2930	40-2946	40-2962	40-2973
		*40-2980	*40-2997	*40-2998	40-3013	40-3022	40-3042	*40-3050	40-3071
		40-3077	40-3080	40-3083	40-3086	40-3089	40-3092	40-3095	40-3101
		40-3110	*42-3181	42-3195	43-3229	*48-3458	48-3468	48-3492	48-3513
		48-3519	48-3522	48-3526	48-3529	49-3561	*49-3572	*49-3586	49-3643
		49-3661	49-3667	50-3724	51-3755	51-3761	51-3828	51-3831	51-3837
		51-3840	51-3843	51-3846	51-3849	51-3852	52-3881	52-3920	*53-3974
		*53-4269	53-4284	53-4290	53-4293	53-4296	53-4302	53-4305	53-4309
		54-4363	54-4372	*55-4477	55-4498	55-4530	55-4539	55-4542	*58-4683
TEMP2	001062 R	#6-250	9-386	10-458	11-576	12-685	12-761	12-767	12-770
		12-777	12-780	12-789	15-882	18-1016	*18-1046	19-1092	26-1437
		26-1510	*26-1604	26-1618	*26-1692	*26-1714	*26-1716	*26-1717	*26-1743
		26-1773	26-1831	26-1834	26-1837	26-1840	26-1846	26-1849	26-1857
		26-1887	26-1891	27-1928	28-2160	29-2268	29-2281	31-2350	31-2351
		*34-2577	*34-2578	*34-2585	*34-2586	*34-2593	*34-2594	34-2608	34-2611
		34-2617	*35-2664	35-2682	*37-2757	37-2760	37-2791	39-2861	39-2907
		40-2983	*40-2999	*40-3003	*40-3004	40-3016	*40-3055	40-3074	40-3077
		40-3080	40-3083	40-3095	*42-3182	42-3195	*48-3463	*48-3464	*48-3488

5-	53	MACRO CALLS
6-	113	LOCAL DATA
7-	171	DISPATCH TABLES
8-	199	LOCAL SYMBOL DEFINITIONS
9-	213	ERROR MESSAGES
11-	378	\$VENA - ENABLE/DISABLE LINE
12-	525	SERBIT - SET/CLE CIRCUIT SERVICE BIT
13-	575	\$VTAD - SET TRIBUTARY STATION ADDRESS
14-	617	\$VACT - SET ACTIVE POLLING RATE
15-	649	\$VDEA - SET DEAD POLLING RATE
16-	681	\$VOWN - SET LINE OWNER
17-	758	\$VCOST - SET LINE COST
18-	805	FNDFR - FIND A FREE CHANNEL
19-	845	FNPID - FIND A PROCESS PDV INDEX
20-	880	FNDLIN - FIND THE DDCMP LINE TABLE FOR A LINE
20-	881	FNDTRI - FIND THE TRIBUTARY TABLE FOR A LINE
21-	944	FNDPCL - FIND THE PCL TRIBUTARY TABLE FOR A LINE
22-	998	FNDMP - FIND THE DMP TRIBUTARY TABLE FOR A LINE
23-	1051	ENALIN - ENABLE A LINE
24-	1084	DSBLIN - DISABLE A LINE
26-	1251	CHKEXE - CHECK IF EXECUTOR NODE
27-	1366	\$VLIN - SET LOOPBACK NODE BLOCK
28-	1471	\$VREM - SET REMOTE NODE BLOCK
29-	1598	FIND PLACE TO INSERT OBJECT BLOCK
30-	1627	FIND PLACE FOR A REMOTE NODE BLOCK
31-	1658	QUICK - FAST REMOTE NODE PROCESSING
32-	1691	QINS - INSERT ENTRY INTO QUEUE
33-	1724	CLRBIT - CLEAR LOOPBACK BIT
34-	1752	SETBIT - SET LOOPBACK BIT
35-	1780	GTSLT - FIND SLT ADDRESS AND STATION NUMBER
36-	1816	\$RPASW - SET/CLEAR RECEIVE PASSWORD
37-	1856	\$TPASW - SET/CLEAR TRANSMIT PASSWORD
38-	1894	\$VERIF - SET VERIFICATION STATE
39-	1928	\$VACP - MOUNT/DISMOUNT THE ACP
40-	2073	\$SMOD - MOUNT/DISMOUNT THE X25ACP
41-	2183	\$VOBJ - SET/CLEAR OBJECT BLOCK
42-	2287	\$VALI - SET/CLEAR ALIAS
43-	2452	VALI - REMOVE ALL OR KNOWN ALIASES
44-	2477	\$VLOG - PROCESS LOGGING REQUEST
45-	2538	CKEVE - CHECK FOR VALID EVENT PARAMETER
46-	2624	PROEVT - PROCESS EVENT MASK FROM REQUEST BLOCK
47-	2666	CKNAME - CHECK FOR VALID SINK NAME
48-	2715	CKMON - CHECK MONITOR SINK NAME
49-	2752	CKCON - CHECK CONSOLE NAME FOR VALIDITY
50-	2784	CKFIL - CHECK FILE SINK NAME
51-	2922	GETLEN - GET LENGTH OF SINK NAME STRING
52-	2955	CKMDEV - CHECK FOR VALID RSX11M DEVICE NAME
53-	3003	CKSINK - CHECK VALIDITY OF SINK NODE ID
54-	3044	CKNODE - CHECK VALIDITY OF NODE ENTITY ID
55-	3075	CHSTA - CHANGE LOGGING STATE
56-	3109	CHFIL - CHANGE EVENT FILTERS
57-	3155	FILPRO - PROCESS FILTER MASKS
58-	3230	CHNAM - CHANGE LOGGING SINK NAME
59-	3291	SETFIL - CREATE FILTER BLOCK
60-	3355	FINDFLT - FIND PLACE TO INSERT FILTER BLOCK
61-	3406	FNDFIL - FIND FILTER BLOCK
62-	3462	FILOPR - OPERATE ON FILTER CONTROL BLOCK

```

435 000436 062600      ADD      (SP)+,R0      ; A WORD INDEX
436 000440 121067      CMPB     (R0),$SLN    ; XPT CHANNEL TABLE CONSISTENT ?
437 000444 001162      BNE       200$      ; IF NE, NO
438
439 000446 012700 017704      MOV      #^RECL,R0      ; Find ECL PDV
440 000452          CALL     $FPDV      ; RET: R0=PDV addr, R1=PDV indx, R2=destroyed
441 000456 103563      BCS       210$      ; If CS, not found - Report as 'ACP not loaded'
442 000460          GETRV     Z,DAT(R0),#N$ELEN ; Read in ECL data base
443 000500 132767 000040 000005G      BITB     #NF$MOU,N$FLG+$BFR ; ACP SHOULD NOT BE LOADED
444 000506 001147      BNE       210$      ; ACP LOADED ???
445 000510 012705 177777      MOV      #-1,R5      ; NONZERO => 'REWRITE DESCRIPTOR'
446
447 000514 016701 000000G      MOV      $SLTA,R1      ; ADDRESS OF SLT
448 000520 032767 000001 000126' 60$ :      BIT      $SER,SERFLG    ; SERVICE ENA/DISA ?
449 000526 001004      BNE       70$      ; IF YES - BRANCH (LINE NEED NOT BE LOADED)
450
451 000530 032761 000000G 000000G      BIT      #LF.RDY,L.FLG(R1) ; IS LINE READY ?
452 000536 001537      BEQ       220$      ; LINE NOT READY !
453
454 000540 032761 000000G 000000G 70$ :      BIT      #LF.BRO,L.FLG(R1) ; ETHERNET LINE ?
455 000546 001402      BEQ       80$      ; IF NOT - BRANCH
456 000550 005000      CLR       R0      ; IF YES - ENABLE ONLY FIRST CIRCUIT
457 000552 000412      BR        90$      ; CONTINUE
458
459 000554 116102 000000G      80$ :      MOVVB    L,NSTA(R1),R2    ; GET NUMBER OF TRIBS
460 000560 001413      BEQ       100$     ; BR IF NOT MULTIPOINT
461 000562 016700 000000G      MOV      $TRIB,R0      ; GET TRIBUTARY WE WANT 'ON'
462 000566 005302      DEC       R2      ; GET TRIBUTARY COUNT - 1
463 000570 020002      CMP      R0,R2      ; DOES THE TRIBUTARY EXIST ?
464 000572 101135      BHI       240$     ; IF H1, NO .. ERROR
465 000574 006300      ASL       R0      ; ELSE, GET WORD OFFSET
466 000576 006300      ASL       R0      ;
467 000600 066700 000000G      90$ :      ADD      $SLTA,R0      ; POINT TO TRIBUTARY INFO.
468 000604 062700 000000G      ADD      #L.MPF,R0      ; ...
469
470 000610 032767 000001 000126' 100$ :      BIT      $SER,SERFLG    ; IS THIS SET CIR SERVICE ?
471 000616 001051      BNE       150$     ; IF YES - BRANCH
472          ;
473          ; IF NO - SET CIR STA
474 000620 032767 000000G 000000G      BIT      #VF.ON,$VFLAG    ; ENABLING ?
475 000626 001423      BEQ       130$     ; NO, DISABLING
476          ;
477          ; ENABLE LINE
478          ;
479 000630 032767 000000G 000000G      BIT      #VF.DLX,$VFLAG    ; DLX ONLY SYSTEM ?
480 000636 001014      BNE       120$     ; BR IF YES
481
482 000640 105761 000000G      TSTB     L,NSTA(R1)      ; MULTIPOINT LINE?
483 000644 001005      BNE       110$     ; BR IF YES
484 000646 032761 000000G 000000G      BIT      #LF.ENA,L.FLG(R1) ; IS CIRCUIT ON?
485 000654 001040      BNE       170$     ; IF EQ, YES
486 000656 000404      BR        120$     ; ELSE TURN CIRCUIT ON
487 000660 132760 000000G 000000G 110$ :      BITB     #SF.ENA,S.FLG(R0) ; IS CIRCUIT ON?
488 000666 001033      BNE       170$     ; BR IF YES
489 000670          120$ :      CALL     ENALIN      ; ELSE, ENABLE THE LINE
490 000674 000424      BR        160$     ; AND CONTINUE
491          ;

```

```

944      .SBTTL FNDPCL - FIND THE PCL TRIBUTARY TABLE FOR A LINE
945
946      **FNDPCL-FIND THE PCL TRIBUTARY TABLE FOR A LINE
947
948      THIS ROUTINE IS CALLED TO FIND THE PCL TRIBUTARY TABLE FOR A LINE.
949
950      INPUTS:
951      $SLTA = SLT ADDRESS
952      $TRIB = TRIBUTARY ADDRESS
953
954      OUTPUTS:
955      C-BIT = ERROR INDICATOR
956      Z-BIT = SEARCH SUCCESS/FAILURE INDICATOR
957      RO-R2 = DESTROYED
958
959      -
960
961      FNDPCL:
962      MOV     $SLTA,R0      ; GET SLT ADDRESS
963      BIT     #LF.RDY,L.FLG(R0) ; IS THIS LINE READY ?
964      BEQ     101$         ; IF EQ, NO .. ERROR
965      MOVB    L.DLC(R0),R2  ; ELSE, GET THE DLC PDV INDEX
966      ADD     $PDVTA,R2    ; POINT INTO THE PDV MAPPING TABLE
967      MOV     (R2),R2       ; POINT TO THE PDV
968      CMP     #*RPCL,Z.NAM(R2) ; IS THIS THE PCL ?
969      BNE     30$          ; IF NE, NO .. ERROR
970      MOV     L.DLS(R0),R1  ; GET THE PCL VIRTUAL ADDRESS
971      CLR     $RBLCK       ; ASSUME UNMAPPED SYSTEM OR DSR ALLOC.
972      TST     .MGMGE       ; IS THIS A MAPPED SYSTEM ?
973      BMI     10$          ; IF MI, NO
974      CMP     #120000,R1   ; ELSE, IS THE LINE TABLE MAPPED ?
975      BHI     10$          ; IF HI, NO
976      MOV     L.DLM(R0),$RBLCK ; ELSE, MAP TO THE PCL PROCESS
977      GETAD   R1,#R.LST+4  ; AND READ IN THE PCL DATA BASE
978      MOV     #$BFR,R2     ; POINT TO I/O BUFFER
979      ADD     #R.LST+2,R2  ; POINT TO THE STATION TABLE LISTHEAD
980      MOV     -(R2),R2     ; GET THE NEXT ENTRY
981      BEQ     102$         ; IF EQ, NOT HERE - ??
982
983      TST     (R2)+         ; ASSUME $RBLCK ALREADY SET UP
984      GETAD   R2,#S.LBE+2  ; SKIP STATION LINK WORD
985      CMPB    $BFR+$LSA,$TRIB ; READ IN THE STATION TABLE
986      BNE     20$          ; IS THIS THE TRIBUTARY ?
987      CLZ     40$          ; IF NE, NO .. KEEP LOOKING
988      BR      40$          ; INDICATE SUCCESS
989      SEZ     30$          ; AND RETURN
990      CLC     40$          ; INDICATE ERROR
991      RETURN      ; INDICATE NO ERROR DETECTED
992
993      ; ERROR CONDITIONS
994
995      101$: ERRPT$ ERR35,$EROUT ; LINE IN WRONG STATE
996      102$: ERRPT$ ERR21,$EROUT ; LINE NOT IN SYSTEM

```



```

1585
1586 005334          RETURN
1587
1588
1589          ; ERROR CONDITIONS
1590
1591 005336          101$: ERRPT$ ERR17,$EROUT      ; INVALID NODE OPTION
1592 005346          102$: ERRPT$ ERR18,$EROUT      ; REMOTE BLOCK ALLOCATION FAILURE
1593 005356          103$: ERRPT$ ERR19,$EROUT      ; NODE NOT IN SYSTEM
1594 005366          104$: ERRPT$ ERR29,$EROUT      ; NODE NAME ALREADY EXISTS
1595
1596          .DSABL  LSB
  
```

```

2042
2043 007254 062767 000001 000000G 48$: ADD #1,$BFR ; POINT TO NEXT BLOCK
2044 007262 PUTRC #2,#2 ; WRITE TO LOCATION 2
2045 007302 016767 000000G 000000G MOV $BFR,$RBLCK ; MAP TO RDB
2046 007310 PUTAD #140000,#2 ; WRITE TO RDB FOR DOUBLE CHECK
2047 007330 000403 BR 100$ ; AND RETURN
2048 007332 000460 46$: BR 111$ ; BRANCH AGAIN
2049
2050 ; DISMOUNT REQUEST
2051
2052 007334 30$: CALL DISMNT ; DISMOUNT THE ACP
2053
2054 007340 100$: RETURN
2055
2056 007342 103$: ERRPT$ ERR8,$EROUT ; ACP NOT IN SYSTEM
2057 007352 104$: ERRPT$ ERR9,$EROUT,FATAL ; POOL EMPTY
2058 007366 105$: ERRPT$ ERR10,$EROUT ; ACP MUST BE FIXED IN FOR RSX11S SYSTEM
2059 007376 010100 107$: MOV R1,R0 ; GET OTHER NTINIT NAME
2060 007400 CALL TSKSCH ; IS IT INSTALLED ?
2061 007404 103406 BCS 1077$ ; NO
2062 007406 1077$: ERRPT$ ERR15,$EROUT,FATAL ; WRONG NTINIT INSTALLED
2063 007422 1077$: ERRPT$ ERR12,$EROUT,FATAL ; NTINIT NOT INSTALLED
2064 007436 108$: ERRPT$ ERR14,$EROUT,FATAL ; ACP NOT FIXED
2065 007452 109$: ERRPT$ ERR16,$EROUT,FATAL ; RSX POOL EMPTY
2066 007466 110$: CALL $DEA16 ; DEALLOCATE NODE
2067 007472 RETURN
2068 007474 111$: CALL DISMNT ; DEALLOCATE EVERYTHING
2069 007500 112$: ERRPT$ ERR13,$EROUT,FATAL ; CANNOT ALLOCATE PLB DATA BASE
2070 007514 112$: ERRPT$ ERR49,$EROUT ; EXEC STA ON ONLY FOR 11-S
2071

```

2624
 2625
 2626
 2627
 2628
 2629
 2630
 2631
 2632
 2633
 2634
 2635
 2636
 2637
 2638 012422 010546
 2639 012424 012700 000000'
 2640 012430 012703 000020
 2641 012434 016705 000002G
 2642 012440
 2643 012454 020527 000740
 2644 012460 103016
 2645 012462 022005
 2646 012464 001007
 2647 012466 012701 000004
 2648 012472 032022
 2649 012474 001007
 2650 012476 005301
 2651 012500 003374
 2652 012502 000405
 2653 012504 062700 000010
 2654 012510 005303
 2655 012512 003363
 2656 012514 000403
 2657 012516 012605
 2658 012520 000241
 2659 012522
 2660
 2661
 2662
 2663 012524 005726
 2664 012526

```

      .SBTTL  PROEVT - PROCESS EVENT MASK FROM REQUEST BLOCK
      ;+
      PROEVT - PROCESS EVENT MASK FROM REQUEST BLOCK
      :
      INPUTS:
      :   R2          - ADDRESS OF EVENT MASK
      :   ROB+LO.CLS - EVENT CLASS
      :
      OUTPUTS:
      :   C-BIT - SUCCESS/FAILURE
      :   R0,R1,R2,R3 DESTROYED
      :
      :-

PROEVT:  MOV    R5, -(SP)          ; SAVE R5
        MOV    #EVTVAL,R0        ; POINT TO VALIDITY TABLE
        MOV    #EVTVSZ,R3        ; AND # OF ENTRIES TO SCAN
        MOV    ROB+LO.CLS,R5     ; GET EVENT CLASS
        ASR$   6,R5              ; MOVE CLASS TO START AT BIT 0
        CMP    R5,#CUSTEV        ; IS THIS A CUSTOMER SPECIFIC EVENT?
        BHS    50$               ; BR IF YES - NO NEED TO FILTER IT
        CMP    (R0)+,R5          ; MATCH ON THIS CLASS?
        BNE    30$               ; IF NE, NO
        MOV    #4,R1             ; PERFORM CHECK ON 4 WORDS
        BIT    (R0)+,(R2)+       ; ANY INVALID BITS PRESENT?
        BNE    40$               ; IF NE, INVALID BITS PRESENT
        DEC    R1                ; CHECK ALL 4 WORDS
        BGT    20$               ; ...
        BR     50$               ; BR IF OK
        ADD    #10,R0            ; POINT TO NEXT ENTRY IN TABLE
        DEC    R3                ; SCAN ALL OF TABLE
        BGT    10$               ; ...
        BR     101$              ; FAILED TO FIND MATCH IN TABLE
        MOV    (SP)+,R5          ; CLEAN UP STACK
        CLC                      ; INDICATE SUCCESS
        60$:  RETURN

      ; ERROR MESSAGES
      101$:  TST    (SP)+          ; CLEAN UP STACK
            ERRPT$ ERR28,$EROUT  ; EVENT CANNOT BE FILTERED
  
```

J 16

```

3155                                     .SBTTL FILPRO - PROCESS FILTER MASKS
3156
3157                                     ;+
3158                                     : FILPRO - PROCESS FILTER MASKS
3159                                     :
3160                                     : INPUTS:
3161                                     : EVQUAL - FILTER CONTROL FLAGS
3162                                     :
3163                                     : OUTPUTS:
3164                                     : EVENT FILTER BLOCK IS OPERATED ON
3165                                     : R3-R5 PRESERVED
3166                                     :
3167                                     :-
3168
3169
3170 014422 004567 000000G FILPRO: JSR R5,$$AVRG ; SAVE R3-R5
3171 014426 016746 000120' MOV EVQUAL,-(SP) ; SAVE FILTER CONTROL FLAGS
3172 014432 042767 000030 000120' BIC #<FF,LIN!FF.ADD>,EVQUAL ; CLEAR QUALIFIER BITS
3173 014440 CALL FNDFIL ; LOOK FOR GLOBAL FILTER BLOCK
3174 014444 012667 000120' MOV (SP)+,EVQUAL ; RESTORE QUALIFIER BITS
3175 014450 010002 MOV R0,R2 ; SAVE ADDRESS OF GLOBAL FILTER BLOCK
3176 014452 001411 BEQ 7$ ; BR IF NOT FOUND
3177 014454 012703 MOV #SBFR+F.SEV,R3 ; POINT TO FILTER MASKS
3178 014460 012704 000076' MOV #TMPMSK,R4 ; POINT TO TEMPORARY STORAGE AREA
3179 014464 012705 000010 MOV #10,R5 ; GET LENGTH OF FILTER MASKS
3180 014470 112324 5$: MOV B (R3)+,(R4)+ ; SAVE FILTER MASKS
3181 014472 005305 DEC R5 ; MORE TO SAVE?
3182 014474 003375 BGT 5$ ; BR IF YES
3183 014476 7$: CALL FNDFIL ; TRY TO FIND EXISTING FILTER BLOCK
3184 014502 103055 40$ BCC 40$ ; BR IF FOUND ONE
3185 014504 032767 000000G 000000G BIT #CM$SET,$CMAND ; SET COMMAND?
3186 014512 001032 BNE 10$ ; BR IF YES
3187 014514 032767 000060 000000G BIT #OP$NOD!OP$LINE,$OPTON ; ANY QUALIFIERS?
3188 014522 001454 BEQ 50$ ; BR IF NO - NOTHING TO DO
3189 014524 005702 TST R2 ; DOES A GLOBAL FILTER BLOCK EXIST?
3190 014526 001452 BEQ 50$ ; FINISHED IF NO GLOBAL FILTER BLOCK
3191
3192 014530 032767 000040 000000G BIT #OP$LINE,$OPTON ; CLE EVE LIN xxx ?
3193 014536 001407 8$ BEQ 8$ ; IF NO - BRANCH
3194 014540 012705 000004G MOV #RQB+LO.EVT,R5 ; GET ADDRESS OF FILTER MASK
3195 014544 005025 CLR (R5)+ ; CLEAR ALL EVENTS
3196 014546 005025 CLR (R5)+ ; ...
3197 014550 005025 CLR (R5)+ ; ...
3198 014552 005025 CLR (R5)+ ; ...
3199 014554 RETURN
3200
3201
3202 014556 012703 000076' 8$: MOV #TMPMSK,R3 ; POINT TO GLOBAL FILTER MASKS
3203 014562 012705 000106' MOV #GMASK,R5 ; GET AREA TO STORE COMPLEMENTED MASKS
3204 014566 CALL SAVCOM ; SAVE COMPELEMENT OF GLOBAL FILTERS
3205 014572 012703 000106' MOV #GMASK,R3 ; POINT TO COMPLEMENTED GLOBAL MASKS
3206 014576 000406 BR 20$ ; CONTINUE
3207
3208 014600 005702 10$: TST R2 ; WAS GLOBAL FILTER BLOCK FOUND?
3209 014602 001411 BEQ 30$ ; BR IF NO
3210 014604 020002 CMP R0,R2 ; OPERATE ON GLOBAL FILTER BLOCK?
3211 014606 001407 BEQ 30$ ; BR IF YES

```

K 16

```

1473      ; DISPLAY STATE AND IDENTIFICATION
1474
1475 007634      60$: CALL   GSTID      ; GET STATE AND IDENTIFICATION
1476 007640      ERRPT$ NMSG3,MSGOUT ; DISPLAY THE STATE AND IDENTIFICATION
1477
1478      ;
1479      ; DISPLAY REMOTE NODES
1480 007650 012701 001062'      70$: MOV   #TEMP2,R1      ; POINT TO TEMP AREA FOR LOOPBACK NODES
1481 007654 005046      CLR   -(SP)      ; COUNT FOR NUMBER OF LOOPBACK NODES
1482 007656 005004      CLR   R4      ; INDICATE HEADER TO BE DISPLAYED
1483
1484 007660      GETRV  $DECPY,#D$END      ; READ DECNET HOME BLOCK
1485 007700 012703 000004G      MOV   #BFR+D$RNN+2,R3 ; STORE REMOTE LISTHEAD
1486
1487 007704      80$: CALL   NXTBLK      ; GET NEXT REMOTE NODE BLOCK IN LIST
1488 007710 103416      BCS   110$      ; BR IF AT END OF LIST
1489 007712 005763 000012      TST   R,FLAG(R3)      ; IS THIS A LOOPBACK NODE?
1490 007716 100404      BMI   90$      ; BR IF YES
1491 007720      CALL   DISNOD      ; DISPLAY THE NODE INFORMATION
1492 007724 005204      INC   R4      ; DON'T DISPLAY HEADER ANY MORE
1493 007726 000766      BR    80$      ; GET NEXT ONE IN LIST
1494
1495      ;
1496      ; FOUND A LOOPBACK NODE - SAVE REMOTE NODE BLOCK TO DISPLAY LATER
1497 007730 010302      90$: MOV   R3,R2      ; POINT TO REMOTE NODE BLOCK
1498 007732 010500      MOV   R5,R0      ; NUMBER OF BYTES TO COPY
1499 007734 112221      100$: MOVB  (R2)+,(R1)+      ; SAVE REMOTE NODE BLOCK
1500 007736 005300      DEC   R0      ; DECREMENT COUNT
1501 007740 003375      BGT   100$      ; BR IF MORE TO COPY
1502 007742 005216      INC   (SP)      ; INCREMENT COUNT
1503 007744 000757      BR    80$      ; GET NEXT ONE IN LIST
1504
1505      ;
1506      ; DISPLAY LOOPBACK NODES IF ANY
1507 007746 012605      110$: MOV   (SP)+,R5      ; GET COUNT OF LOOPBACK NODES
1508 007750 001534      BEQ   200$      ; BR IF NONE TO DISPLAY
1509 007752 005004      CLR   R4      ; INDICATE HEADER TO BE PRINTED
1510 007754 012703 001062'      MOV   #TEMP2,R3      ; POINT TO REMOTE NODE BLOCK
1511 007760      120$: CALL   DISLOO      ; DISPLAY LOOP NODE INFORMATION
1512 007764 005204      INC   R4      ; INDICATE HEADER NOT TO BE PRINTED
1513 007766 062703 000014      ADD   #R.LEN,R3      ; POINT TO NEXT BLOCK
1514 007772 005305      DEC   R5      ; DECREMENT COUNT
1515 007774 003371      BGT   120$      ; BR IF MORE TO DISPLAY
1516 007776 000521      BR    200$      ; FINISHED
1517
1518      ;
1519      ; SHOW SPECIFIC NODE
1520 010000 012700 000411'      130$: MOV   #NSPEC,R0      ; SHOW SPECIFIC NODE
1521 010004 112021      140$: MOVB  (R0)+,(R1)+      ; STORE STRING
1522 010006 001376      BNE   140$      ; ...
1523
1524 010010      GETRV  $DECPY,#D$END      ; READ DECNET HOME BLOCK
1525 010030 016767 000014G 171166      MOV   #BFR+D$LNUM,TEMP4 ; STORE EXECUTOR ADDRESS
1526 010036 012702 000000G      MOV   #R0B,R2      ; POINT TO INPUT PARAMETER BLOCK
1527 010042 032767 000000G 000000G      BIT   #VF.ADD,#VFLAG      ; DID USER SPECIFY A NODE ADDRESS?
1528 010050 001015      BNE   150$      ; BR IF YES
1529 010052 026267 000012 000006G      CMP   LR.NAM(R2),$BFR+D$LNAM ; IS THIS THE EXECUTOR NODE?

```

```

2139          ,SBTTL $DLIN - SHOW LINE
2140
2141          ;+
2142          ; $DLIN - SHOW KNOWN LINES
2143          ; SHOW ACTIVE LINES
2144          ; SHOW LINE X
2145
2146          ; INPUTS: NONE
2147
2148          ; OUTPUTS: THE LINE INFORMATION IS DISPLAYED ON THE USER'S TERMINAL
2149
2150          ; ALL REGISTERS ARE DESTROYED
2151          ; -
2152
2153          $DLIN:: CALL CKTYPE          ; GET COMMAND TYPE
2154          BCC 10$                     ; BR IF SUCCESS
2155          10$: ERRPT$ SYNERR,$EROUT   ; ELSE SYNTAX ERROR
2156          MOV $SLTMA,R5              ; GET ADDRESS OF SLT ADDRESS TABLE
2157          MOV $SLTNM,R3              ; GET NUMBER OF SLT'S IN SYSTEM
2158          MOV #TEMP2,R1             ; STORAGE FOR HEADER MESSAGE
2159          TSTB $QUAL                 ; SHOW SPECIFIC LINE?
2160          BEQ 70$                     ; BR IF YES
2161          CMPB #QF$KNO,$QUAL         ; SHOW KNOWN LINES
2162          BNE 20$                     ; BR IF NO
2163          JMP 160$                    ; BR IF YES
2164
2165          ; SHOW ACTIVE LINES
2166
2167          20$: MOV #LACTIV,R0          ; MUST BE SHOW ACTIVE LINES
2168          30$: MOVB (R0)+,(R1)+        ; STORE THE HEADER MESSAGE
2169          BNE 30$                     ;
2170          CLR R4                      ; INDICATE HEADER TO BE DISPLAYED
2171          40$: CALL NXTLIN            ; GET NEXT SYSTEM LINE TABLE IN SYSTEM
2172          BCC 50$                     ; BR IF FOUND ONE
2173          JMP 190$                     ; BR IF NO MORE
2174          50$: CMP Z,NAM(R1),#*RDLM   ; IS THIS DLM?
2175          BEQ 40$                     ; BR IF YES - NOT REALLY A LINE
2176          BIT #LF,ENA,L.FLG(R0)      ; IS LINE ACTIVE?
2177          BEQ 40$                     ; BR IF NO
2178          60$: CALL DISLIN           ; DISPLAY LINE INFORMATION
2179          INC R4                      ; INDICATE HEADER NOT TO BE DISPLAYED
2180          BR 40$                      ; GET NEXT LINE IN SYSTEM
2181
2182          ; SHOW SPECIFIC LINE
2183
2184          70$: MOV #LSPEC,R0           ; GET STRING FOR HEADER MESSAGE
2185          80$: MOVB (R0)+,(R1)+        ; STORE STRING
2186          BNE 80$                     ;
2187          CLR R4                      ; INDICATE HEADER TO BE DISPLAYED
2188          90$: CALL NXTLIN            ; GET NEXT LINE IN SYSTEM
2189          BCC 187$                     ; BR IF NO MORE
2190          CMP Z,NAM(R1),#*RDLM        ; IS THIS DLM?
2191          BEQ 90$                     ; BR IF YES - NOT REALLY A LINE
2192          CMP Z,NAM(R1),LR.LIN+R0B    ; IS THIS A MATCH?
2193          BNE 90$                     ; BR IF NO
2194
2195          000000G 015160 103004
2196          000000G 016705 000000G
2197          000000G 016703 000000G
2198          001062' 012701 001062'
2199          000000G 105767 000000G
2200          001434 015216 001434
2201          000004 122767 000000G
2202          001002 015226 001002
2203          000164 015230 000164
2204
2205          000441' 015234 012700 000441'
2206          0112021 015240 112021
2207          001376 015242 001376
2208          005004 015244 005004
2209          0103002 015246 0103002
2210          0000204 015252 103002 0000204
2211          0000000G 015254 000167 0000000G 015355
2212          0026127 015260 026127
2213          001767 015266 001767
2214          0032760 015270 032760 002000 000000
2215          001763 015276 001763
2216          005204 015300 005204
2217          000757 015306 000757
2218
2219          000472' 015310 012700 000472'
2220          0112021 015314 112021
2221          001376 015316 001376
2222          005004 015320 005004
2223          0103454 015322 0103454
2224          000000G 015330 026127 000000G 015355
2225          001771 015336 001771
2226          000000G 015340 026167 000000G 000002G
2227          001365 015346 001365

```

2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723

021126 012703 000000G
021132 012705 000122
021136
021142 103430
021144 032767 000400 000000G
021152 001020
021154 126767 000017G 000046G
021162 001365
021164 012700 000047G
021170 012701 000020G
021174 016702 000046G
021200 006202
021202 005502
021204 122021
021206 001353
021210
021214 152767 000004 160352
021222 000241
021224

.SBTTL NXTDTE - GET NEXT DTE BLOCK

;

NXTDTE - FIND NEXT DTE BLOCK TO OPERATE ON

INPUTS:

\$BFR - ADDRESS OF NEXT DTE DESCRIPTOR
\$OPTON - OPTIONS WORD
LO.DTE+RQB - DTE ADDRESS TO LOOK FOR

OUTPUTS:

CARRY CLEAR:
\$BFR CONTAINS NEXT DTE BLOCK
CARRY SET:
END OF DTE DESCRIPTOR LIST

NXTDTE: MOV \$BFR,R3 ; GET ADDRESS OF POINTER TO NEXT DTE
MOV #L\$LEN,R5 ; GET LENGTH OF DTE DESCRIPTOR
10\$: CALL NXTBLK ; READ IN NEXT BLOCK
BCS 40\$; BR IF END OF LIST
BIT #OP\$KDT,\$OPTON ; KNOWN DTE SPECIFIED?
BNE 30\$; BR IF YES - HAVE A MATCH
CMPB L\$DTEL+\$BFR,LO.LDT+RQB ; SAME LENGTH?
BNE 10\$; BR IF NO - TRY AGAIN
MOV #LO.DTE+RQB,R0 ; POINT TO SPECIFIED DTE ADDRESS
MOV #L\$DTEA+\$BFR,R1 ; POINT TO DTE ADDRESS IN DESCRIPTOR
MOV LO.LDT+RQB,R2 ; GET NUMBER OF DIGITS IN ADDRESS
ASR R2 ; MAKE IT A BYTE COUNT
ADC R2
20\$: CMPB (R0)+,(R1)+ ; DO WE HAVE A MATCH?
BNE 10\$; BR IF NO - TRY AGAIN
SOB R2,20\$
30\$: BISB #F.FND,FLAGS ; INDICATE FOUND ONE
CLC ; INDICATE SUCCESS
40\$: RETURN

```

3236 .SBTTL FNDBLK - Find next block meeting given criteria [TD001]
3237 + [TD001]
3238 *** FNDBLK - Find next block meeting given criteria [TD001]
3239 [TD001]
3240 INPUTS [TD001]
3241 R0 - Address of buffer containing field to be matched. [TD001]
3242 Buffer contains one-byte pre-count followed by data. [TD001]
3243 If R0 is zero, the next block in sequence is returned. [TD001]
3244 R1 - Length of each block to be read in search. [TD001]
3245 R2 - Offset in block of field to be matched against buffer [TD001]
3246 (not used if no matching required) [TD001]
3247 R3 - Offset in block of one-byte count of bytes in field [TD001]
3248 to be matched (R3 >= 0); negated length of field in [TD001]
3249 block (R3 < 0) [TD001]
3250 $BFR - The first word of $BFR contains a pointer to the [TD001]
3251 first block to be considered. [TD001]
3252 [TD001]
3253 OUTPUTS [TD001]
3254 If carry clear: [TD001]
3255 $BFR - Contains the first matching block [TD001]
3256 R0 - Address of matched field in buffer [TD001]
3257 R1 - Actual length of matched field in buffer [TD001]
3258 If carry set: [TD001]
3259 No block found [TD001]
3260 [TD001]
3261 Registers R2-R5 preserved [TD001]
3262 [TD001]
3263 NOTE [TD001]
3264 The rather contorted parameters are necessary to take account [TD001]
3265 both of old-format and extended PSI data structures. [TD001]
3266 [TD001]
3267 R3 is used as follows: [TD001]
3268 If R3 is positive, the count at this offset in the block must [TD001]
3269 match the count in the buffer addressed by R0 and all (R3) characters [TD001]
3270 at offset (R2) must match (R3) characters at address (R0+1). [TD001]
3271 If R3 is negative, the count in the match buffer is ignored, [TD001]
3272 and (-R3) characters are checked for a match regardless of the [TD001]
3273 count at address (R0). [TD001]
3274 [TD001]
3275 FNDBLK: [TD001]
3276 JSR R5,$SAVRG ; Save registers R2-R5 [TD001]
3277 MOV $BFR,R5 ; Get head-of-chain address [TD001]
3278 BEQ 99$ ; If zero, then end-of-chain [TD001]
3279 CEACCS$ R5 ; Map to head-of-chain [TD001]
3280 GETAD R5,R1 ; Read in required length of data [TD001]
3281 TST R0 ; Check if match required [TD001]
3282 BEQ 50$ ; If zero, any block will do [TD001]
3283 TST R3 ; Check whether length explicit [TD001]
3284 BPL 10$ ; If positive, length to be determined [TD001]
3285 MOV R3,R5 ; Length specified by caller [TD001]
3286 NEG R5 ; Make length positive [TD001]
3287 BR 20$ [TD001]
3288 MOV $BFR(R3),R5 ; Get length from data block [TD001]
3289 CMPB R5,(R0) ; Length must equal match string length [TD001]
3290 BNE 5$ ; If not same, must find another block [TD001]
3291 MOV R3,-(SP) ; Save R3 [TD001]
3292 MOV R0,R3 ; Get address of data in match string [TD001]
3293 [TD001]
3275 024620 004567 000000G
3276 024620 004567 000000G
3277 024624 016705 000000G
3278 024630 001452
3279 024632
3280 024642
3281 024656 005700
3282 024660 001423
3283 024662 005703
3284 024664 100003
3285 024666 010305
3286 024670 005405
3287 024672 000404
3288 024674 116305 000000G
3289 024700 120510
3290 024702 001350
3291 024704 010346
3292 024706 010003

```



```

3791 027164 026727 152012 001750 50$: CMP TEMP3,#1000. ; IS ADDRESS THREE DIGITS
3792 027172 002005 BGE 60$ ; IF NO - BRANCH
3793 027174 ERRPT$ DNMSG4,MSGOUT ; PRINT AREA.ADDRESS
3794 027204 000444 BR 110$ ; CONTINUE
3795
3796 027206 60$: ERRPT$ DNMS10,MSGOUT ; PRINT AREA.ADDRESS
3797 027216 000437 BR 110$ ; CONTINUE
3798
3799 027220 026727 151756 000012 70$: CMP TEMP3,#10. ; IS ADDRESS ONE DIGIT ?
3800 027226 002005 BGE 80$ ; IF NO - BRANCH
3801 027230 ERRPT$ DNMSG6,MSGOUT ; PRINT AREA.ADDRESS
3802 027240 000426 BR 110$ ; CONTINUE
3803
3804 027242 026727 151734 000144 80$: CMP TEMP3,#100. ; IS ADDRESS TWO DIGITS ?
3805 027250 002005 BGE 90$ ; IF NO - BRANCH
3806 027252 ERRPT$ DNMSG5,MSGOUT ; PRINT AREA.ADDRESS
3807 027262 000415 BR 110$ ; CONTINUE
3808
3809 027264 026727 151712 001750 90$: CMP TEMP3,#1000. ; IS ADDRESS THREE DIGITS ?
3810 027272 002005 BGE 100$ ; IF NO - BRANCH
3811 027274 ERRPT$ DNMSG3,MSGOUT ; PRINT AREA.ADDRESS
3812 027304 000404 BR 110$ ; CONTINUE
3813
3814 027306 100$: ERRPT$ DNMSG7,MSGOUT ; PRINT AREA.ADDRESS
3815
3816 027316 110$: RESRG <R2,R1> ; RESTORE R1
3817 027322 RETURN
3818
3819 .ENABL LC
3820
3821 ;
3822 ; DISPLAY MESSAGES
3823 ;
3824 027324 ERMSG$ <%NRemote%NNode Name%N>
3825 027351 ERBLK$ DNMSG1
3826
3827 027354 ERMSG$ < %D %A>
3828 027364 ERBLK$ DNMSG2,<TEMP3,TEMP1>
3829
3830 027374 ERMSG$ <%D.%D %A>
3831 027405 ERBLK$ DNMSG3,<TEMP6,TEMP3,TEMP1>
3832
3833 027416 ERMSG$ < %D.%D %A>
3834 027430 ERBLK$ DNMSG4,<TEMP6,TEMP3,TEMP1>
3835
3836 027442 ERMSG$ <%D.%D %A>
3837 027454 ERBLK$ DNMSG5,<TEMP6,TEMP3,TEMP1>
3838
3839 027466 ERMSG$ <%D.%D %A>
3840 027501 ERBLK$ DNMSG6,<TEMP6,TEMP3,TEMP1>
3841
3842 027512 ERMSG$ <%D.%D %A>
3843 027522 ERBLK$ DNMSG7,<TEMP6,TEMP3,TEMP1>
3844
3845 027534 ERMSG$ < %D.%D %A>
3846 027550 ERBLK$ DNMSG8,<TEMP6,TEMP3,TEMP1>
3847

```

```

4375 .SBTTL DISLIN - DISPLAY LINE INFORMATION
4376
4377
4378
4379
4380
4381
4382
4383
4384
4385
4386
4387
4388
4389
4390
4391
4392
4393
4394
4395 033074
4396 033110 005704
4397 033112 001010
4398 033114
4399 033124 016600 000012
4400 033130 016601 000010
4401
4402
4403
4404 033134
4405 033140
4406 033150 016600 000012
4407 033154 016601 000010
4408
4409
4410
4411 033160 012705 001062'
4412 033164 012704 000567'
4413 033170 105760 000014
4414 033174 001407
4415 033176 032760 000400 000000
4416 033204 001436
4417 033206 012704 000652'
4418 033212 000433
4419 033214 012704 000621'
4420 033220 032760 000020 000000
4421 033226 001025
4422 033230 012704 000645'
4423 033234 026127 000000G 043340
4424 033242 001417
4425 033244 026127 000000G 043341
4426 033252 001413
4427 033254 116001 000003
4428 033260 066701 000000G
4429 033264 011101
4430 033266 026127 000000G 045452
4431 033274 001402

;+
DISLIN - DISPLAY LINE INFORMATION
;
; INPUTS:
; R0 = LINE'S SLT ADDRESS
; R1 = LINE'S DDM PDV ADDRESS
; R3 = NUMBER OF SLT'S REMAINING TO SCAN
; R4 = 0 PRINT HEADER INFORMATION ALONG WITH LINE INFORMATION
; ELSE, PRINT ONLY LINE INFORMATION
;
; OUTPUTS:
; LINE INFORMATION IS DISPLAYED ON USER'S TERMINAL
;
; ALL REGISTERS ARE PRESERVED
;-
DISLIN: SAVRG <R0,R1,R2,R3,R4,R5> ; SAVE R0-R5
; TST R4 ; DISPLAY HEADER?
; BNE 10$ ; BR IF NO
; ERRPT$ LMSG1,MSGOUT ; DISPLAY HEADER MESSAGE
; MOV 12(SP),R0 ; RESTORE R0
; MOV 10(SP),R1 ; RESTORE R1
;
; DISPLAY LINE-ID
10$: CALL FMTLIN ; CREATE THE LINE-ID
; ERRPT$ LMSG2,MSGOUT ; DISPLAY THE LINE-ID
; MOV 12(SP),R0 ; RESTORE R0
; MOV 10(SP),R1 ; RESTORE R1
;
; GET LINE TYPE
;
; MOV #TEMP2,R5 ; STORAGE FOR LINE TYPE
; MOV #LCNTRL,R4 ; ASSUME IT IS CONTROLLER
; TSTB L,NSTA(R0) ; MULTIPPOINT LINE?
; BEQ 15$ ; IF NO - BRANCH
; BIT #LF,BRO,L.FLG(R0) ; ETHERNET DEVICE ?
; BEQ 20$ ; IF NOT - BRANCH
; MOV #BROAD,R4 ; IF YES - SET UP TYPE DISPLAY
; BR 20$ ; CONTINUE
; MOV #LTRIB,R4 ; ASSUME TRIBUTARY
; BIT #LF,MTP,L.FLG(R0) ; MULTIPPOINT LINE?
; BNE 20$ ; BR IF YES
; MOV #LLAB,R4 ; ASSUME X25
; CMP Z,NAM(R1),#*RKMX ; IS THIS A KMX LINE?
; BEQ 20$ ; BR IF YES
; CMP Z,NAM(R1),#*RKMY ; IS THIS A KMY LINE?
; BEQ 20$ ; BR IF YES
; MOVB L,DLC(R0),R1 ; GET DLC'S PDV
; ADD $PDVTA,R1 ; INDEX INTO VECTOR TABLE
; MOV (R1),R1 ; GET PDV ADDRESS
; CMP Z,NAM(R1),#*RLAB ; IS IT LAPB?
; BEQ 20$ ; BR IF YES

```

```

4858                                     .S8TTL CUNDEC - CONVERT CUG NUMBER TO DECIMAL
4859
4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871 035300                                CUNDEC: SAVRG    <R0,R1,R2>          ; SAVE R0,R1,R2
4872 035306 016702 000010G                MOV    G$CUG+$8FR,R2      ; GET CUG NUMBER
4873 035312 010200                        MOV    R2,R0          ; GET CUG NUMBER
4874 035314 042700 177417                  BIC    #*C<360>,R0      ; ISOLATE FIRST DIGIT
4875 035320 006200                        ASR    R0              ; ...
4876 035322 006200                        ASR    R0              ; ...
4877 035324 006200                        ASR    R0              ; ...
4878 035326 006200                        ASR    R0              ; ...
4879
4880 035330 012701 001750                  MOV    #1000.,R1        ; UNIT IS THOUSANDS
4881 035334                        CALL   $MUL                ; GET BINARY VALUE
4882 035340 010167 143660                  MOV    R1,TEMP4        ; STORE THOUSANDS DIGIT
4883 035344 010200                        MOV    R2,R0          ; GET CUG NUMBER
4884 035346 042700 177760                  BIC    #*C<17>,R0      ; ISOLATE SECOND DIGIT
4885 035352 012701 000144                  MOV    #100.,R1        ; UNIT IS HUNDREDS
4886 035356                        CALL   $MUL                ; GET BINARY VALUE
4887 035362 060167 143636                  ADD    R1,TEMP4        ; UPDATE BINARY VALUE
4888 035366 010200                        MOV    R2,R0          ; GET CUG NUMBER
4889 035370 000300                        SWAB   R0              ; MOVE DIGIT TO LOW BYTE
4890 035372 042700 177417                  BIC    #*C<360>,R0      ; ISOLATE THIRD DIGIT
4891 035376 006200                        ASR    R0              ; ...
4892 035400 006200                        ASR    R0              ; ...
4893 035402 006200                        ASR    R0              ; ...
4894 035404 006200                        ASR    R0              ; ...
4895
4896 035406 012701 000012                  MOV    #10.,R1         ; UNIT IS TENS
4897 035412                        CALL   $MUL                ; GET BINARY VALUE
4898 035416 060167 143602                  ADD    R1,TEMP4        ; STORE TENS DIGIT
4899 035422 010200                        MOV    R2,R0          ; GET CUG NUMBER
4900 035424 042700 170377                  BIC    #*C<7400>,R0    ; ISOLATE FOURTH DIGIT
4901 035430 006200                        ASR    R0              ; ...
4902 035432 006200                        ASR    R0              ; ...
4903 035434 006200                        ASR    R0              ; ...
4904 035436 006200                        ASR    R0              ; ...
4905 035440 006200                        ASR    R0              ; ...
4906 035442 006200                        ASR    R0              ; ...
4907 035444 006200                        ASR    R0              ; ...
4908 035446 006200                        ASR    R0              ; ...
4909
4910 035450 060067 143550                  ADD    R0,TEMP4        ; STORE ONES DIGIT
4911 035454                        RESRG  <R2,R1,R0>          ; RESTORE REGISTERS
4912 035462                        RETURN

```

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 3 K 8
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
DSMS1	023340 R	40-2936 #40-3071
DSMS10	024050 R	40-2966 #40-3104
DSMS11	024076 R	40-2969 #40-3107
DSMS12	024122 R	40-2977 #40-3110
DSMS2A	023406 R	40-3005 #40-3074
DSMS2B	023456 R	40-2991 40-3001 #40-3077
DSMS2C	023524 R	40-2989 #40-3080
DSMS3	023576 R	40-3018 #40-3083
DSMS4	023622 R	40-3024 #40-3086
DSMS5	023652 R	40-3046 #40-3089
DSMS6	023704 R	40-3053 #40-3092
DSMS7	023742 R	40-3056 #40-3095
DSMS8	024006 R	40-3065 #40-3098
DSMS9	024026 R	40-2951 #40-3101
DTMS1	021000 R	35-2642 #35-2679
DTMS2	021070 R	35-2665 #35-2682
DTMS3	021124 R	35-2673 #35-2685
D\$ACL	000027	40-2971
D\$AMXC	000072	26-1692
D\$AMXH	000074	26-1693
D\$ANN	000000	10-464
D\$CUGL	000023	40-2955 40-3033
D\$CUGN	000022	40-3037 40-3039
D\$DATL	000030	40-3010
D\$DELF	000045	26-1727
D\$DELOW	000046	26-1738
D\$DSL	000022	40-2953 40-3031 41-3146
D\$DST	000010	41-3141
D\$DTL	000031	40-3020
D\$END	= 000104	10-461 18-1023 22-1298 25-1397 26-1439 26-1484 26-1524 26-1605 26-1684
		26-1706 26-1713 53-4208 58-4682 59-4743
D\$FLEN	000032	41-3140
D\$GLEN	000032	41-3144
D\$GVBL	000032	40-2952 40-3030 41-3145
D\$HOST	000022	18-1026 26-1606 26-1607
D\$INAC	000044	26-1725
D\$INCT	000042	26-1721
D\$IPL	000051	26-1748
D\$IID	000020	59-4745
D\$LNAM	000006	18-1031 18-1032 18-1033 26-1529 26-1531 26-1533 26-1610 58-4685
D\$LNUM	000014	18-1027 18-1030 26-1525 26-1537 26-1607 58-4683
D\$LST	000047	26-1746
D\$MAXC	000064	26-1685
D\$MAXH	000066	26-1707
D\$MAXV	000070	26-1708
D\$NA	000062	26-1690
D\$NAM	000004	40-3003 40-3004
D\$NBEA	000056	26-1698
D\$NBRA	000054	26-1697
D\$NN	000060	26-1686
D\$NOD	000010	40-2945
D\$OBJ	000003	40-2981 40-2999

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 16 K 9
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
TEMP3	001202 R	48-3522 48-3529 49-3575 49-3576 49-3592 49-3658 49-3664 49-3667 50-3707 50-3724 50-3730 53-3978 53-4160 53-4216 *53-4270 53-4284 53-4290 53-4293 53-4296 53-4302 53-4305 54-4363 55-4411 *55-4482 55-4530 55-4536 55-4539 55-4545 58-4687 59-4751 *6-251 7-317 *12-708 17-988 *17-990 17-993 *17-995 18-1017 18-1052 *26-1606 26-1633 *26-1634 *26-1685 *26-1693 *26-1708 *26-1718 *26-1719 *26-1745 *26-1746 26-1846 26-1849 26-1853 26-1857 26-1863 26-1878 26-1881 26-1887 26-1891 *31-2355 *31-2356 31-2379 *34-2572 *34-2573 *34-2579 *34-2580 *34-2587 *34-2588 *34-2595 *34-2596 34-2608 34-2611 34-2614 34-2617 35-2682 *49-3579 *49-3587 49-3667 *50-3701 50-3727 *51-3760 *51-3770 51-3781 51-3786 51-3791 51-3799 51-3804 51-3809 51-3828 51-3831 51-3834 51-3837 51-3840 51-3843 51-3846 51-3849 51-3852 52-3920 *53-4092 53-4115 53-4287 53-4299 53-4302 54-4340 54-4366 55-4443 *55-4491 55-4533 55-4536 55-4539 56-4575 57-4627 TEMP4 001224 R *6-252 22-1301 *22-1303 22-1305 22-1309 22-1314 *26-1525 26-1554 *26-1686 *26-1697 *26-1707 *26-1720 *26-1721 *26-1747 *26-1748 26-1853 26-1860 26-1863 26-1887 26-1891 *31-2353 31-2379 37-2791 *63-4882 *63-4887 *63-4898 *63-4910 TEMP5 001244 R *6-253 *26-1722 *26-1723 26-1887 TEMP6 001554 R *6-254 *26-1724 *26-1725 26-1887 *51-3769 *51-3771 *51-3776 *51-3776 *51-3776 *51-3776 *51-3776 *51-3776 *51-3776 *51-3776 *51-3776 *51-3776 *51-3776 *51-3776 *51-3834 51-3837 51-3840 51-3843 51-3846 51-3849 51-3852 51-3859 51-3862 51-3865 TEMP7 001556 R *6-255 12-770 12-773 *18-1059 *18-1069 *26-1468 *26-1592 *26-1642 *26-1655 *26-1698 *26-1733 *26-1734 26-1837 26-1849 26-1860 26-1881 26-1887 26-1887 TEMP8 001560 R *6-256 26-1672 26-1866 TF.FFE = 000004 *5-88 TF.ROF = 000001 *5-88 TF.RTO = 000002 *5-88 TF.TAE = 000010 *5-88 TF.TOF = 000004 *5-88 TF.TTO = 000010 *5-88 TF.URE = 000001 *5-88 TF.USA = 000002 *5-88 TIATT 001616 R *6-267 43-3228 TIDET 001646 R *6-268 8-344 9-424 9-427 10-534 10-537 11-627 11-631 12-752 26-1564 26-1823 27-2127 27-2130 28-2230 28-2233 TR.ALF 000102 *5-88 TR.DEV 000100 *5-88 TR.LEN = 000210 *5-88 53-4049 TR.MST 000074 *5-88 TR.PRE 000076 *5-88 TR.TLZ 000072 *5-88 TSKSCH 035052 R 39-2864 59-4716 #60-4778 TX.NXM = 000002 *5-88 TX.OVR = 000001 *5-88 TX.TMO = 000004 *5-88 TY\$CHA = 000400 *5-99 43-3224 TY\$EVE = 001000 *5-99 43-3217 TY\$STA = 002000 *5-99 TY\$SUM = 000000 *5-99 TS\$FLAG 000044 *5-76 TS\$IF 000013 *5-76 TS\$IFL 000013 *5-76 TS\$IFO 000013 *5-76

63-	3647	GETNXT - GET NEXT FILTER BLOCK
64-	3711	SETEVT - SET THE EVENT FILTER MASKS
65-	3732	CLREVT - CLEAR THE SPECIFIED EVENT MASKS
66-	3757	CHKEVT - CHECK THE SPECIFIED EVENT MASKS
67-	3787	SAVFIL - SAVE FILTER MASKS
68-	3809	SAVCOM - SAVE COMPLEMENT OF FILTER MASKS
69-	3835	STREVT - STORE THE EVENT MASKS
70-	3856	INIEVT - INITIALIZE THE EVENT FILTER MASKS
71-	3876	SETQLF - SET QUALIFIER FIELDS IN FILTER BLOCK
72-	3916	DEAFIL - DEALLOCATE FILTER BLOCK
73-	3944	STQUAL - SET QUALIFIER BITS
74-	3988	ALPLB - ALLOCATE PLB DATA BASE
75-	4077	DISMNT - DISMOUNT THE ACP
76-	4126	XSDSMT - DISMOUNT THE X25ACP
77-	4147	FNDDCB - FIND A DCB ADDRESS
78-	4170	\$VHST - SET/CLEAR HOST ADDRESS
79-	4192	\$VLTM - SET/CLEAR LISTEN TIMER
80-	4212	\$VHTM - SET/CLEAR HELLO TIMER
81-	4232	FNDCTR - FIND COUNTER BLOCK
82-	4287	RUN - RUN A TASK
83-	4358	CHKXPT - CHECK XPT CONTEXT
84-	4400	TSKSCH - SEARCH FOR TASK TCB
85-	4429	\$FPDV - FIND A PDV ADDRESS

```

492 ; DISABLE LINE
493
494 000676 032767 000000G 000000G 130$: BIT #VF.DLX,$VFLAG ; DLX ONLY SYSTEM ?
495 000704 001013 BNE 140$ ; BR IF YES
496 000706 032761 000000G 000000G BIT #LF.ENA,L.FLG(R1) ; IS CIRCUIT ALREADY OFF?
497 000714 001420 BEQ 170$ ; IF EQ, YES
498 000716 105761 000000G TSTB L.NSTA(R1) ; MULTIPOINT LINE?
499 000722 001404 BEQ 140$ ; BR IF NO
500 000724 132760 000000G 000000G BITB #SF.ENA,S.FLG(R0) ; IS CIRCUIT ALREADY OFF?
501 000732 001411 BEQ 170$ ; BR IF YES
502 000734 140$: CALL DSBLIN ; ELSE, DISABLE THE LINE
503 000740 000402 BR 160$ ; CONTINUE
504
505 000742 150$: CALL SERBIT ; SET/CLE SERVICE BIT
506
507 000746 005705 160$: TST R5 ; REWRITE DESCRIPTOR?
508 000750 001402 BEQ 170$ ; IF EQ THEN NO
509 000752 PUTRC ; REWRITE DESCRIPTOR
510 000756 042767 000000G 000000G 170$: BIC #VF.DLX,$VFLAG ; CLEAR DLX ONLY BIT - NO LONGER NEEDED
511 000764 RETURN
512
513
514 000766 180$: ERRPT$ ERR1,$EROUT,FATAL ; CHANNEL NOT IN XPT'S DATA BASE
515 001002 190$: ERRPT$ ERR2,$EROUT ; LINE NOT ASSIGNED TO XPT
516 001012 200$: ERRPT$ ERR5,$EROUT,FATAL ; CHANNEL TABLE INCONSISTENT WITH REV TAB
517 001026 210$: ERRPT$ ERR6,$EROUT ; ACP LOADED
518 001036 032767 000000G 000000G 220$: BIT #VF.ON,$VFLAG ; TURNING LINE ON ?
519 001044 001404 BEQ 230$ ; NO
520 001046 ERRPT$ ERR7,$EROUT ; LINE NOT LOADED
521 001036 230$: ERRPT$ ERR7,$EROUT ;
522 001066 240$: ERRPT$ ERR21,$EROUT ; TRIB DOES NOT EXIST

```

```

998                                     .SBTTL FNDDMP - FIND THE DMP TRIBUTARY TABLE FOR A LINE
999
1000                                **--FNDDMP-FIND THE DMP TRIBUTARY TABLE FOR A LINE
1001                                THIS ROUTINE IS CALLED TO FIND THE DMP TRIBUTARY TABLE FOR A LINE.
1002                                INPUTS:
1003                                $SLTA = SLT ADDRESS
1004                                $TRIB = TRIBUTARY ADDRESS
1005                                OUTPUTS:
1006                                C-BIT = ERROR INDICATOR
1007                                Z-BIT = SEARCH SUCCESS/FAILURE INDICATOR
1008                                RO-R2 = DESTROYED
1009                                -
1010
1011
1012
1013
1014
1015 FNDDMP:
1016 002732 016700 000000G MOV $SLTA,R0 ; GET SLT ADDRESS
1017 002736 032760 000000G 000000G BIT #LF.RDY,L.FLG(R0) ; IS THIS LINE READY ?
1018 002744 001463 BEQ 101$ ; IF EQ, NO .. ERROR
1019 002746 116002 000000G MOV L.DLC(R0),R2 ; ELSE, GET THE DLC PDV INDEX
1020 002752 066702 000000G ADD $PDVTA,R2 ; POINT INTO THE PDV MAPPING TABLE
1021 002756 011202 MOV (R2),R2 ; POINT TO THE PDV
1022 002760 022762 015430 000000G CMP #*RDMP,Z.NAM(R2) ; IS THIS THE DMP?
1023 002766 001047 BNE 30$ ; IF NE, NO .. ERROR
1024 002770 016001 000000G MOV L.DLS(R0),R1 ; GET THE PCL VIRTUAL ADDRESS
1025 002774 005067 000000G CLR $RBLCK ; ASSUME UNMAPPED SYSTEM OR DSR ALLOC.
1026 003000 005767 000000G TST .MGMGE ; IS THIS A MAPPED SYSTEM ?
1027 003004 100406 BMI 10$ ; IF MI, NO
1028 003006 022701 120000 CMP #120000,R1 ; ELSE, IS THE LINE TABLE MAPPED ?
1029 003012 101003 BHI 10$ ; IF HI, NO
1030 003014 016067 000000G 000000G MOV L.DLM(R0),$RBLCK ; ELSE, MAP TO THE PCL PROCESS
1031 003022 10$: GETAD R1,#P.LEN ; AND READ IN THE DMP DATA BASE
1032 003040 012702 000000G MOV #BFR,R2 ; POINT TO I/O BUFFER
1033 003044 062702 000014 ADD #P.TRIB,R2 ; POINT TO THE STATION TABLE LISTHEAD
1034 003050 011202 20$: MOV (R2),R2 ; GET THE NEXT ENTRY
1035 003052 001424 BEQ 102$ ; IF EQ, NOT HERE - ??
1036
1037 003054 GETAD R2,#T.TRLG ; ASSUME $RBLCK ALREADY SET UP
1038 003072 126767 000005G 000000G CMPE $BFR+T.TRLN,$TRIB ; READ IN THE STATION TABLE
1039 003100 001363 BNE 20$ ; IS THIS THE TRIBUTARY ?
1040 003102 000244 CLZ ; IF NE, NO .. KEEP LOOKING
1041 003104 000401 BR 40$ ; INDICATE SUCCESS
1042 003106 000264 30$: SEZ ; AND RETURN
1043 003110 000241 40$: CLC ; INDICATE ERROR
1044 003112 RETURN ; INDICATE NO ERROR DETECTED
1045
1046 ; ERROR CONDITIONS
1047
1048 003114 101$: ERRPT$ ERR35,$EROUT ; LINE IN WRONG STATE
1049 003124 102$: ERRPT$ ERR21,$EROUT ; LINE NOT IN SYSTEM

```


K 13

```

1598                                     .SBTTL  FIND PLACE TO INSERT OBJECT BLOCK
1599                                     ;+
1600                                     *** - FINOBJ - FIND PLACE TO INSERT OBJECT
1601                                     ;
1602                                     INPUTS:
1603                                     R1 = OBJECT TYPE NUMBER TO BE INSERTED
1604                                     ;
1605                                     OUTPUTS:
1606                                     R0 = PRESERVED
1607                                     R1 = UNMAPPED ADDRESS OF PREVIOUS BLOCK
1608                                     R2 = DESTROYED
1609                                     ;
1610                                     -
1611                                     FINOBJ:
1612 005376 012702 000000G      MOV      #OBJHDL,R2      ; GET START OF OBJECTS LIST
1613 005402 011267 000000G      MOV      (R2),BFR      ; GET FIRST OBJECT IN LIST
1614 005406 010146      MOV      R1,-(SP)      ; SAVE OBJECT NUMBER ON STACK
1615 005410 010267 000164'      MOV      R2,CURUNM      ; INITIALIZE CURRENT UNMAPPED ADDRESS
1616 005414 016767 000164' 000166' 10$:      MOV      CURUNM,PRVUNM      ; SAVE UNMAPPED ADDRESS OF PREVIOUS
1617 005422 016767 000000G 000164'      MOV      BFR,CURUNM      ; GET NEXT OBJECT IN LIST
1618 005430 001417      BEQ      30$      ; IF EQ, NO MORE
1619 005432      CEACC$      CURUNM,R2      ; CONVERT TO MAPPED ADDRESS
1620 005444      GETAD      R2,#0,LEN      ; READ OBJECT IN INTERNAL BUFFER
1621 005462 121667 000002G      CMPB     (SP),BFR+Q.TYP      ; DOES OBJECT GO HERE ?
1622 005466 101352      BHI      10$      ; IF HI, NO
1623 005470 016701 000166'      MOV      PRVUNM,R1      ; RETRIEVE UNMAPPED ADDRESS OF PREVIOUS
1624 005474 005726      TST      (SP)+      ; CLEAN UP STACK
1625 005476      RETURN
  
```

L 13

```

2073          .SBTIL $SMOD - MOUNT/DISMOUNT THE X25ACP
2074      +
2075      **-$SMOD- MOUNT/DISMOUNT THE X25ACP
2076
2077      INPUTS:
2078          $VFLAG = TRANSITORY FLAGS
2079
2080      OUTPUTS:
2081          X25ACP IS MOUNTED/DISMOUNTED OR ERROR CONDITION IS REPORTED
2082      -
2083
2084      $SMOD::
2085          007524 012700 055430      MOV    #*RNW,RO      ; GET NW'S PROCESS NAME
2086          007530      CALL    $FPDV      ; IS IT IN SYSTEM?
2087          007534 103001      BCC    1$      ; BR IF YES
2088          007536      RETURN
2089
2090          007540 032767 000000G 000000G 1$:  BIT    #VF.SS,$VFLAG      ; S-SYSTEM ?
2091          007546 001005      BNE    5$      ; IF YES - BRANCH      ;[MP01]
2092          007550      ERRPT$ ERR49,$EROUT      ; NO MOD X25-S STA ON FOR M/M+ ;[MP01]
2093
2094          007560      RETURN
2095
2096          007562 010167 000072'      5$:  MOV    R1,PDVX      ; SAVE PDV INDEX
2097          007566 012700 000004'      MOV    #X25NAM,RO      ; POINT AT RAD50 ACP NAME
2098          007572      CALL    TSKSCH      ; GET TCB ADDRESS
2099          007576 103566      BCS    103$      ; TASK NOT IN SYSTEM !
2100
2101          .IF NDF R$$MPL
2102          007600 032767 002000 000034G      BIT    #T2.FXD,T.ST2+$BFR      ; ACP FIXED IN MEMORY ?      ;[MP01]
2103          007606 001574      BEQ    105$      ; IF EQ, NO
2104          .ENDC
2105
2106          007610 012700 000014'      MOV    #NTI11S,RO      ; ASSUME 11S
2107          007614 012701 000010'      MOV    #NTINAM,R1      ; PRESET FOR ERROR
2108          007620 032767 000000G 000000G      BIT    #VF.SS,$VFLAG      ; IS THIS AN S-SYSTEM ?
2109          007626 001003      BNE    10$      ; YES
2110          007630 010046      MOV    RO,-(SP)      ; SWAP NAMES
2111          007632 010100      MOV    R1,RO      ; ...
2112          007634 012601      MOV    (SP)+,R1      ; ...
2113          007636      10$:  CALL    TSKSCH      ; GET TCB ADDRESS
2114          007642 103562      BCS    107$      ; TASK NOT INSTALLED
2115          007644 032767 000000G 000000G      BIT    #VF.SS,$VFLAG      ; S-SYSTEM ?
2116          007652 001404      BEQ    15$      ; IF NOT - BRANCH
2117          .IF NDF R$$MPL
2118          007654 032767 002000 000034G      BIT    #T2.FXD,T.ST2+$BFR      ; IS TASK FIXED      ;[MP01]
2119          007662 001572      BEQ    108$      ; TASK NOT FIXED
2120          .ENDC
2121
2122      ; QUEUE SEND DATA TO NTINIT
2123
2124          007664 016705 000000G      15$:  MOV    $RADDR,R5      ; TCB ADDRESS
2125          007670 012701 000044      MOV    #36.,R1      ; SIZE OF SEND DATA I/O PACKET
2126
2127      ; CHECK IF MOUNT OR DISMOUNT
2128
2129          007674 032767 000000G 000000G      BIT    #VF.ON,$VFLAG      ; MOUNT PASS TO STRT TIMERS
  
```

```

2666 .SBTTL CKNAME - CHECK FOR VALID SINK NAME
2667
2668
2669
2670 CKNAME - CHECK FOR VALID SINK NAME
2671
2672
2673 INPUTS:
2674 $CMAND - COMMAND TYPE
2675 $OPTON - OPTIONS WORD
2676 $QUAL - COMMAND QUALIFIER WORD
2677 RQB+LO.SNM - ASCII SINK NAME
2678
2679 OUTPUTS:
2680 CARRY SUCCESS/FAILURE
2681
2682 CKNAME:
2683 012536 032767 000000G 000000G BIT #CM$SET,$CMAND ; IS THIS A SET COMMAND?
2684 012544 001005 BNE 10$ ; BR IF YES
2685 012546 105767 000035G TSTB RQB+LO.SNM ; WAS A NAME SPECIFIED?
2686 012552 000241 CLC ; ASSUME NO NAME SPECIFIED
2687 012554 001437 BEQ 60$ ; BR IF NO - OK
2688 012556 000443 BR 102$ ; ELSE ERROR
2689 012560 032767 000004 000000G 10$: BIT #Q$SKNO,$QUAL ; IS THIS FOR KNOWN LOGGING?
2690 012566 001037 BNE 102$ ; BR IF YES - ERROR
2691 012570 012701 000134' 20$: MOV #FILNAM,R1 ; POINT TO SINK NAME STORAGE
2692 012574 012702 000064 MOV #MXFIL+MXCON+MXMON,R2 ; GET MAXIMUM LENGTH OF SINK NAME
2693 012600 105021 30$: CLR (R1)+ ; INITIALIZE SINK NAME
2694 012602 005302 DEC R2 ; MORE TO INITIALIZE?
2695 012604 003375 BGT 30$ ; BR IF YES
2696 012606 032767 000100 000000G BIT #OP$MON,$OPTON ; IS THIS FOR LOGGING MONITOR?
2697 012614 001404 BEQ 40$ ; BR IF NO
2698 012616 CALL CKMON ; CHECK FOR VALID MONITOR NAME
2699 012622 103014 BCC 60$ ; AND EXIT IF SUCCESSFUL
2700 012624 000414 BR 101$ ; ELSE BAD SINK NAME
2701 012626 032767 000400 000000G 40$: BIT #OP$CON,$OPTON ; IS THIS FOR LOGGING CONSOLE?
2702 012634 001404 BEQ 50$ ; BR IF NO
2703 012636 CALL CKCON ; CHECK FOR VALID CONSOLE NAME
2704 012642 103004 BCC 60$ ; AND EXIT IF SUCCESSFUL
2705 012644 000404 BR 101$ ; ELSE BAD SINK NAME
2706 012646 50$: CALL CKFIL ; ELSE MUST BE LOGGING FILE
2707 012652 103401 BCS 101$ ; BR IF INVALID SINK NAME
2708 012654 60$: RETURN
2709
2710 : ERRORS
2711
2712 012656 101$: ERRPT$ ERR39,$EROUT ; INVALID SINK NAME
2713 012666 102$: ERRPT$ ERR42,$EROUT ; INVALID PARAMETER GROUPING

```

```

3212 014610 012703 000076'      MOV      #TMPMSK,R3      ; POINT TO GLOBAL FILTER BLOCK MASKS
3213 014614 012705 000004G      20$: MOV      #RGB+LO.EVT,R5    ; POINT TO FILTER MASKS
3214 014620                      CALL     CLREVT      ; CLEAR ANY MASKS SET IN GLOBAL BLOCK
3215 014624 103413              BCS      50$          ; BR IF NO WORK TO DO
3216 014626                      30$: CALL     SETFIL      ; CREATE A FILTER BLOCK
3217 014632 103010              BCC      50$          ; BR IF FILTER BLOCK CREATED
3218 014634 000410              BR       101$         ; BR IF ERROR
3219
3220 014636                      40$: CALL     FILOPR      ; PERFORM OPERATION ON FILTER BLOCK
3221 014642 103004              BCC      50$          ; BR IF UNNECESSARY TO DEALLOCATE BLOCK
3222 014644 012705 000000G      MOV      #BFR,R5      ; POINT TO FILTER BLOCK TO DEALLOCATE
3223 014650                      CALL     DEAFIL      ; DEALLOCATE THE FILTER BLOCK
3224 014654                      50$: RETURN
3225
3226                      ; ERROR MESSAGES
3227
3228 014656                      101$: ERRPT$ ERR38,$EROUT ; UNABLE TO ALLOCATE FILTER BLOCK

```

```

1530 010060 001017 BNE 160$ ; BR IF NO
1531 010062 026267 000014 000010G CMP LR.NAM+2(R2),$BFR+D$LNAM+2 ;
1532 010070 001013 BNE 160$ ; BR IF NO
1533 010072 026267 000016 000012G CMP LR.NAM+4(R2),$BFR+D$LNAM+4 ;
1534 010100 001007 BNE 160$ ; BR IF NO
1535 010102 000470 BR 220$ ; ELSE DO SHOW EXECUTOR
1536 010104 150$:
1537 010104 026267 000012 000014G CMP LR.ADD(R2),$BFR+D$LNAM ; IS THIS THE EXECUTOR NODE?
1538 010112 001464 BEQ 220$ ; BR IF YES
1539 010114 005713 TST (R3) ; ANY NODES IN SYSTEM?
1540 010116 001453 BEQ 210$ ; BR IF NO
1541 010120 160$: CALL NXTBLK ; GET NEXT REMOTE NODE NAME BLOCK
1542 010124 103450 BCS 10$ ; BR IF NOT IN SYSTEM
1543 010126 032767 000000G 000000G BIT #VF.ADD,$VFLAG ; DID USER SPECIFY ADDRESS?
1544 010134 001015 BNE 170$ ; BR IF YES
1545 010136 026263 000012 000002 CMP LR.NAM(R2),R.NAM(R3) ; IS THIS THE ONE WE ARE LOOKING FOR?
1546 010144 001365 BNE 160$ ; BR IF NO
1547 010146 026263 000014 000004 CMP LR.NAM+2(R2),R.NAM+2(R3) ;
1548 010154 001361 BNE 160$ ; BR IF NO
1549 010156 026263 000016 000006 CMP LR.NAM+4(R2),R.NAM+4(R3) ;
1550 010164 001355 BNE 160$ ; BR IF NO
1551 010166 000404 BR 180$ ; THIS IS THE ONE. DISPLAY IT
1552 010170 026263 000012 000010 170$: CMP LR.ADD(R2),R.ADD(R3) ; IS THIS THE ONE WE ARE LOOKING FOR?
1553 010176 001350 BNE 160$ ; BR IF NO
1554 010200 026267 000012 171016 180$: CMP LR.ADD(R2),TEMP4 ; IS THIS THE EXECUTOR NODE?
1555 010206 001426 BEQ 220$ ; BR IF YES
1556 010210 ERRPT$ NMSG1,MSGOUT ; DISPLAY HEADER MESSAGE
1557 010220 005004 CLR R4 ; INDICATE DISPLAY COLUMN HEADER
1558 010222 005763 000012 TST R.FLAG(R3) ; IS IT A LOOPBACK NODE?
1559 010226 100003 BPL 190$ ; BR IF NO
1560 010230 CALL DISLOO ; DISPLAY LOOPBACK NODE INFORMATION
1561 010234 000402 BR 200$ ; FINISHED
1562 010236 190$: CALL DISNOD ; DISPLAY THE NODE INFORMATION
1563 010242 000167 001546 200$: JMP 510$ ; FINISHED
1564 010246 210$: DIR$ #TIDET ; DETACH TERMINAL
1565 010254 ERRPT$ NERR1,$EROUT ; NODE NOT IN SYSTEM
1566 ;
1567 ; SHOW EXECUTOR
1568 ;
1569 010264 220$: ERRPT$ NMSG1,MSGOUT ; DISPLAY HEADER
1570 ;
1571 ; MAKE CHECK FOR DLX ONLY -- PROCESSING WILL BE DIFFERENT IF TRUE
1572 ;
1573 010274 012700 015370 MOV #*RDLX,R0 ; LOOK FOR DLX PROCESS
1574 010300 CALL FNPDV ; CAN WE FIND PDV ?
1575 010304 103407 BCS 230$ ; NO - MUST NOT BE DLX ONLY
1576 010306 012700 114224 MOV #*RXPT,R0 ; CHECK FOR XPT
1577 010312 CALL FNPDV ; IS PDV THERE ?
1578 010316 103002 BCC 230$ ; YES - NOT DLX ONLY - CONTINUE
1579 010320 000167 001434 JMP 500$ ; NO - DLX ONLY SYSTEM
1580 ;
1581 ; DISPLAY EXECUTOR ADDRESS AND NAME
1582 ;
1583 010324 230$: CALL GTXEC ; GET EXECUTOR ADDRESS AND NAME
1584 ;
1585 010330 016701 170500 MOV TEMP1,R1 ; COPY EXECUTOR ADDRESS
1586 010334 042767 000000G 170472 BIC #ARAMSK,TEMP1 ; STRIP OFF AREA BITS

```

```

2196                                     ; MATCH ON CONTROLLER NUMBER
2197
2198
2199 015350 032767 000400 000000G      BIT    #WC.CNT,$QUAL      ; WILD CARD CONTROLLER?
2200 015356 001004                BNE    110$                ; BR IF YES, MATCH
2201 015360 126067 000012 000004G      CMPB   L.CTL(R0),LR.CTL+RQB    ; IS THIS A MATCH?
2202 015366 001355                BNE    90$                ; BR IF NO
2203
2204                                     ; MATCH ON UNIT NUMBER
2205
2206 015370 032767 001000 000000G 110$: BIT    #WC.UNIT,$QUAL      ; WILD CARD UNIT?
2207 015376 001004                BNE    120$                ; BR IF YES, MATCH
2208 015400 126067 000013 000005G      CMPB   L.UNIT(R0),LR.UNIT+RQB    ; IS THIS A MATCH?
2209 015406 001345                BNE    90$                ; BR IF NO
2210 015410                CALL   DISLIN      ; DISPLAY LINE INFORMATION
2211 015414 005204                INC    R4                ; INDICATE HEADER TO BE DISPLAYED
2212 015416 000741                BR     90$                ; CONTINUE SCAN
2213
2214                                     ; SHOW KNOWN LINES
2215
2216 015420 012700 000456' 160$: MOV    #LKNOWN,R0      ; GET STRING FOR HEADER MESSAGE
2217 015424 112021 170$: MOVB   (R0)+,(R1)+      ; STORE STRING
2218 015426 001376                BNE    170$
2219 015430 005004                CLR    R4
2220 015432 180$: CALL   NXTLIN      ; INDICATE HEADER TO BE DISPLAYED
2221 015436 103412 190$: BCS    190$      ; GET NEXT LINE IN SYSTEM
2222 015440 026127 000000G 015355      CMP    Z.NAM(R1),#^RDLM    ; BR IF END OF TABLE
2223 015446 001771                BEQ    180$      ; IS THIS DLM?
2224 015450                CALL   DISLIN      ; BR IF YES - NOT REALLY A LINE
2225 015454 005204                INC    R4                ; DISPLAY LINE INFORMATION
2226 015456 000765                BR     180$      ; INDICATE HEADER NOT TO BE DISPLAYED
2227 015460 005704 187$: TST    R4                ; GET NEXT LINE
2228 015462 001404                BEQ    200$      ; WAS THE SPECIFIED LINE FOUND?
2229
2230 015464 190$: DIR$    #TIDET      ; DETACH TERMINAL
2231 015472                RETURN      ; FINISHED
2232
2233 015474 200$: DIR$    #TIDET      ; DETACH TERMINAL
2234 015502      ERRPT$  LIERR1,$EROUT    ; LINE NOT IN SYSTEM
2235
2236      .ENABL   LC
2237
2238      ; ERROR MESSAGES
2239
2240 015512      ERMSG$  < Line not in system>
2241 015535      ERLK$    LIERR1

```

```

2725          .SBTIL DMGRP - SHOW MODULE X25-PROTOCOL GROUPS
2726
2727          ;+
2728          : DMGRP - SHOW MODULE X25-PROTOCOL WITH GROUP QUALIFIER
2729
2730          : INPUTS:
2731          : $OPTON - OPTIONS WORD
2732          : LO.GRO+RGB - SPECIFIED GROUP NAME
2733
2734          : OUTPUTS:
2735          : GROUP INFORMATION DISPLAYED ON USER'S TERMINAL
2736
2737          :
2738
2739 021226 016700 000000G DMGRP: MOV $PSIPT,R0 ; POINT TO PSI HOME BLOCK
2740 021232 062700 000010 ADD #H$CUG,R0 ; POINT TO CUG LISTHEAD
2741 021236 GETRV R0,#2 ; READ IN LISTHEAD
2742
2743 021254 10$: CALL NXTGRP ; Get next group block to operate on
2744 021260 103475 BCS 60$ ; Branch if done
2745 021262 012702 001034' MOV #TEMP1,R2 ; Point to storage for group name
2746 021266 112022 20$: MOVB (R0)+(R2)+ ; Store group name using length and
2747 021270 SOB R1,20$ ; ... address returned by NXTGRP
2748 021274 020227 001034' 30$: CMP R2,#TEMP1 ; Delete trailing spaces
2749 021300 101404 BLOS 35$ ; ...
2750 021302 124227 000040 CMPB -(R2),#SPACE
2751 021306 001772 BEQ 30$
2752 021310 005202 INC R2
2753 021312 105022 35$: CLRB (R2)+ ; Create ASCIZ string
2754 021314 ERRPT$ GRMS1,MSGOUT ; Display group name
2755
2756 021324 ;
2757 021330 CALL CUNDEC ; CONVERT CUG NUMBER TO DECIMAL
2758 021334 005067 157526 CLR TEMP2 ; ASSUME GROUP IS NOT BILATERAL
2759 021342 001406 000001 000014G BIT #GF$BUG,G$FLG+$BFR ; IS IT A BILATERAL TYPE?
2760 021344 012700 001062' BEO 50$ ; BR IF NO
2761 021350 012701 000750' MOV #TEMP2,R0 ; POINT TO STORAGE FOR TYPE STRING
2762 021354 112120 40$: MOV #GRPBIL,R1 ; POINT TO ASCIZ STRING
2763 021356 001376 BNE 40$ ; STORE ASCIZ STRING
2764 021360 016746 000000G 50$: MOV $BFR,-(SP) ; SAVE POINTER TO NEXT CUG BLOCK
2765 021364 016700 000012G MOV $DTE+$BFR,R0 ; POINT TO ADDRESS OF DTE DESCRIPTOR
2766 021370 CEACCS$ R0 ; CONVERT TO MAPPED ADDRESS
2767 021400 GETAD R0,#L$LEN ; READ IN DTE DESCRIPTOR
2768 0214 012701 000020G MOV #BFR+$DTEA,R1 ; POINT TO DTE ADDRESS
2769 0214 116702 000017G MOVB $BFR+$DTEL,R2 ; GET NUMBER OF DIGITS IN ADDRESS
2770 021426 012703 001034' MOV #TEMP1,R3 ; POINT TO OUTPUT BUFFER
2771 021432 CALL BCDASC ; CONVERT TO ASCIZ
2772 021436 012667 000000G MOV (SP)+,$BFR ; RESTORE POINTER TO NEXT CUG BLOCK
2773 021442 ERRPT$ GRMS2,MSGOUT ; DISPLAY INFO TO USER'S TERMINAL
2774 021452 000700 BR 10$ ; TRY NEXT BLOCK
2775
2776 021454 132767 000004 160112 60$: BITB #F.FND,FLAGS ; FOUND A CUG BLOCK?
2777 021462 001015 BNE 80$ ; BR IF YES
2778 021464 032767 000100 000000G BIT #OP$GRP,$SOPTON ; SHOW SPECIFIC GROUP?
2779 021472 001005 BNE 70$ ; BR IF YES
2780 021474 ERRPT$ NOINFO,MSGOUT ; DISPLAY NO INFORMATION MESSAGE
2781 021504 000404 BR 80$ ; AND EXIT

```

```

3293 024710 005203 INC R3 ; [TD001]
3294 024712 012704 MOV #BFR,R4 ; Get address of data in buffer ; [TD001]
3295 024716 060204 ADD R2,R4 ; [TD001]
3296 024720 CALL MATCH ; Look for a match ; [TD001]
3297 024724 012603 MOV (SP)+,R3 ; Restore length indicator ; [TD001]
3298 024726 103736 BCS S$ ; If no match, must find another block ; [TD001]
3299 ; [TD001]
3300 ; Return actual length and address of matched field ; [TD001]
3301 ; [TD001]
3302 024730 010301 50$: MOV R3,R1 ; Prepare to return length of name ; [TD001]
3303 024732 100002 BPL 60$ ; If positive, need to look in block ; [TD001]
3304 024734 005401 NEG R1 ; If negative, magnitude is length ; [TD001]
3305 024736 000402 BR 70$ ; [TD001]
3306 024740 116101 60$: MOVB BFR(R1),R1 ; Get actual length from block ; [TD001]
3307 024744 012700 70$: MOV #BFR,R0 ; Address of start of block ; [TD001]
3308 024750 060200 ADD R2,R0 ; Add offset to give address of name ; [TD001]
3309 024752 000241 CLC ; Signal block found ; [TD001]
3310 024754 RETURN ; [TD001]
3311 ; [TD001]
3312 024756 000261 99$: SEC ; No block found ; [TD001]
3313 024760 RETURN ; [TD001]
3314 ; [TD001]
3315 ; + [TD001]
3316 ; *** MATCH - Compare byte strings ; [TD001]
3317 ; [TD001]
3318 ; INPUTS ; [TD001]
3319 ; R3, R4 - Addresses of strings to compare ; [TD001]
3320 ; R5 - Length of strings to compare ; [TD001]
3321 ; [TD001]
3322 ; OUTPUTS ; [TD001]
3323 ; Carry clear if strings match ; [TD001]
3324 ; Registers R3-R5 destroyed ; [TD001]
3325 ; [TD001]
3326 024762 MATCH: R5 ; Check if strings empty ; [TD001]
3327 024764 005705 TST R5 ; If so, this is a match ; [TD001]
3328 024766 001404 BEQ 10$ ; [TD001]
3329 024770 001004 5$: CMPB (R3)+,(R4)+ ; Compare bytes until mismatch ; [TD001]
3330 024772 005305 BNE 20$ ; [TD001]
3331 024774 003374 DEC R5 ; [TD001]
3332 024776 000241 BGT 5$ ; [TD001]
3333 025000 10$: CLC ; All bytes match ; [TD001]
3334 025002 20$: RETURN ; No match ; [TD001]
3335 025004 RETURN ; [TD001]

```


3848 027562
3849 027575
3850
3851 027606
3852 027617
3853
3854
3855

ERMSG\$ < %D.%D %A>
ERBLK\$ DNMSG9,<TEMP6,TEMP3,TEMP1>

ERMSG\$ < %D.%D %A>
ERBLK\$ DNMS10,<TEMP6,TEMP3,TEMP1>

.DSABL LC

```

DISLIN - DISPLAY LINE INFORMATION

4432 033276 012704 000605'      20$:  MOV    #LPOINT,R4      ; ELSE IT MUST BE POINT-TO-POINT
4433 033302 112425              MOVVB  (R4)+,(R5)+      ; STORE LINE TYPE STRING
4434 033304 001376              BNE    20$              ; ...
4435
4436              ; GET LINE OWNER
4437
4438 033306 016600 000012      MOV    12(SP),R0      ; RESTORE R0
4439 033312 016601 000010      MOV    10(SP),R1      ; RESTORE R1
4440 033316 116003 000021      MOVVB  L,NMST+1(R0),R3      ; ASSUME OWNER IS IN LINE SLT
4441 033322 005703              TST    R3              ; IS THERE AN OWNER SET?
4442 033324 001430              BEQ    50$              ; BR IF NO
4443 033326 012700 001202'      MOV    #TEMP3,R0      ; STORAGE FOR LINE OWNER
4444 033332 066703 000000G      ADD    $PDVTA,R3      ; INDEX INTO PDV TABLE
4445 033336 011303              MOV    (R3),R3      ; GET PDV ADDRESS
4446 033340 016301 000000G      MOV    Z,NAM(R3),R1      ; GET OWNER NAME IN RAD50
4447 033344              CALL    $CSTA      ; CONVERT IT TO ASCII
4448 033350 124027 000040      CMPB   -(R0),#SPACE      ; DELETE TRAILING BLANKS
4449 033354 001004              BNE    30$              ; ...
4450 033356 124027 000040      CMPB   -(R0),#SPACE      ; ...
4451 033362 001001              BNE    30$              ; ...
4452 033364 005300              DEC    R0              ; ...
4453 033366 112760 000000 000001 30$:  MOVVB  #0,1(R0)      ; CREATE ASCII STRING
4454 033374              ERRPT$  LMSG3,MSGOUT      ; PRINT TYPE, OWNER
4455 033404 000040              BR      60$              ; CONTINUE WITH CSR
4456 033406              50$:  ERRPT$  LMSG9,MSGOUT      ; DISPLAY TYPE
4457
4458              ; GET CONTROLLER CSR, VECTOR, PRIORITY
4459
4460 033416 016600 000012      60$:  MOV    12(SP),R0      ; RESTORE R0
4461 033422 016601 000010      MOV    10(SP),R1      ; RESTORE R1
4462 033426 005760 000016      TST    L,KRBA(R0)      ; IS LINE LOADED?
4463 033432 001447              BEQ    80$              ; BR IF NO
4464 033434 012703 000000G      MOV    #SBFR,R3      ; GET ADDRESS OF AREA TO READ IN KRB
4465
4466              .IF DF R$$MPL                      ;[MP01]
4467              MOV    L,KRBA(R0),R0      ; GET KRB ADDRESS      ;[MP01]
4468              ADD    #K.PRM,R0      ; GET ACTUAL KRB START  ;[MP01]
4469              GETRV  R0,#KRBLEN      ; READ KRB      ;[MP01]
4470              SUB    #K.PRM,R3      ; ADD K.PRM (K.PRM IS NEGATIVE);[MP01]
4471
4472              .IFF
4473 033440              GETRV  L,KRBA(R0),#K.CSR+2      ; GET KRB ADDRESS      ;[MP01]
4474
4475              .ENDC
4476
4477 033460 016367 000002 145346      MOV    K.CSR(R3),TEMP1      ; GET CSR
4478 033466 005046              CLR    -(SP)      ; USE SCRATCH WORD ON STACK
4479 033470 156316 000001              BISB  K,VCT(R3),(SP)      ; GET VECTOR/4, DON'T SIGN EXTEND
4480 033474 006316              ASL    (SP)      ; MULTIPLY BY 4 TO GET VECTOR
4481 033476 006316              ASL    (SP)      ; ...
4482 033500 012667 145356              MOV    (SP)+,TEMP2      ; GET VECTOR
4483 033504 005000              CLR    R0      ; INIT REG
4484 033506 156300 000000              BISB  K,PRI(R3),R0      ; GET PRIORITY
4485 033512 000241              CLC      ; DON'T SIGN EXTEND
4486 033514 106000              RORB  R0      ; GET PRIORITY IN LOW THREE BITS
4487 033516 106200              ASRB  R0      ; ...
4488 033520 106200              ASRB  R0      ; ...

```

```

4914
4915
4916
4917
4918
4919
4920
4921
4922
4923
4924
4925
4926
4927
4928
4929
4930
4931
4932 035464 005005
4933 035466 152105
4934 035470 010546
4935      000004
4936
4937
4938 035502 062716 000060
4939 035506 112623
4940 035510 005302
4941 035512 003411
4942 035514 010546
4943 035516 042716 177760
4944 035522 062716 000060
4945 035526 112623
4946 035530 005302
4947 035532 003354
4948 035534 105023
4949 035536

```

```

.SBTTL BCDASC - CONVERT BCD TO ASCII

+
BCDASC - CONVERT BCD TO ASCII
:
: INPUTS:
: R1 - ADDRESS OF BCD VALUE TO CONVERT
: R2 - LENGTH OF BCD VALUE (NUMBER OF DIGITS)
: R3 - NEXT AVAILABLE BYTE IN OUTPUT BUFFER
:
: OUTPUTS:
: R3 - UPDATED
:
: REGISTERS MODIFIED
: R1,R2,R5
:
-
BCDASC: CLR R5 ; INITIALIZE DIGIT
        BISB (R1)+,R5 ; GET TWO DIGITS
        MOV R5,-(SP) ; COPY THEM
        .REPT 4
        ASR (SP) ; GET HIGH FOUR BITS
        .ENDR
        ADD #60,(SP) ; CONVERT TO ASCII
        MOVB (SP)+,(R3)+ ; STORE DIGIT IN OUTPUT BUFFER
        DEC R2 ; DECREMENT COUNT
        BLE 10$ ; BR IF FINISHED
        MOV R5,-(SP) ; GET DIGITS
        BIC #*(<17>,(SP) ; ISOLATE LOW FOUR BITS (SECOND DIGIT)
        ADD #60,(SP) ; CONVERT TO ASCII
        MOVB (SP)+,(R3)+ ; STORE SECOND DIGIT IN OUTPUT BUFFER
        DEC R2 ; DONE?
        BGT BCDASC ; BR IF NO
        CLRB (R3)+ ; CREATE ASCII STRING
10$: RETURN

```

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 4 L 8
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
D\$OUTT	000043	26-1723
D\$PAL	000026	40-2967
D\$PRI	000002	40-2980 40-2998
D\$PRL	000024	40-2958 40-2984
D\$RETF	000050	26-1719
D\$RNN	000002	22-1299 25-1398 26-1440 26-1485 53-4209
D\$SEG	000036	26-1714
D\$SHI	000020	40-3051 40-3055
D\$SLO	000016	40-3050 40-3051
D\$USL	000025	40-2960
D\$VBL	000032	40-2996
D.LEN	= 000036	#7-294 31-2354 49-3578 62-4850
D.NAM	000004	31-2355 49-3579 62-4851
D.UCB	000002	49-3581 62-4853
D.UCBL	000010	49-3582
D.UNIT	000006	49-3584
ENABLE	000477 R	#6-202 53-4161
EVENT	000000 R	#6-116 43-3216
EVTMSK	= 001034 R	#7-314 7-315 *19-1088 *19-1089 *19-1090 *19-1091 19-1113 19-1115 19-1117
		19-1119 *20-1189 *20-1190 *20-1191 *20-1192 *20-1193 *20-1194 *20-1195 *20-1196
		*20-1197 *20-1198 *20-1199 *20-1200 24-1377
E\$NBR	000014	#5-76
E\$NBS	000020	#5-76
E\$NCR	000034	#5-76
E\$NCS	000036	#5-76
E\$NIC	000044	#5-76
E\$NLEN	000050	#5-76
E\$NLLA	000012	#5-76
E\$NLNK	000000	#5-76
E\$NML	000040	#5-76
E\$NMR	000024	#5-76
E\$NMS	000030	#5-76
E\$NNOD	000002	#5-76
E\$NRT	000042	#5-76
E\$NRTP	000005	#5-76
E\$NSEG	000010	#5-76
E\$NTIM	000046	#5-76
E\$NUSE	000004	#5-76
E\$STRT	000006	#5-76
FF.ADD	= 000020	#5-84 5-84 19-1097
FF.CIR	= 000040	#5-84 5-84
FF.CON	= 000001	#5-84 13-825
FF.FIL	= 000002	#5-84 13-829
FF.HST	= 040000	#5-84
FF.LIN	= 000010	#5-84 5-84 19-1093
FF.MOD	= 000100	#5-84 5-84
FF.MON	= 000004	#5-84 13-821
FF.MSK	= 000077	#5-84 12-728 19-1102 19-1109
FF.PRT	= 000200	#5-84 5-84
FF.QL	= 000370	#5-84
FF.REM	= 100000	#5-84 17-979 17-986
FLAGS	001574 R	#6-258 *19-1126 19-1138 *19-1143 *19-1144 19-1148 *19-1155 23-1344 *23-1350

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 17 L 9
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
T\$LIFS	000013	#5-76
T\$LIN	000000	#5-76 53-4265
T\$LIPS	000006	#5-76
T\$LLD	000012	#5-76
T\$LLDC	000045	#5-76
T\$LLDL	000012	#5-76
T\$LLDO	000012	#5-76
T\$LLDS	000012	#5-76
T\$LLEN	000046	#5-76 53-4262 53-4267
T\$LOPR	000002	#5-76
T\$LTCL	000024	#5-76
T\$LTIM	000026	#5-76
T\$LTIPR	000014	#5-76
T\$LTIPS	000020	#5-76
T\$NAPL	000004	#5-76
T\$NFE	000000	#5-76
T\$NLEN	000010	#5-76
T\$NNI	000002	#5-76
T\$NOPL	000006	#5-76
T\$NRNI	000042	#5-76
T\$NRPL	000005	#5-76
T\$NRUL	000007	#5-76
T\$NVR	000001	#5-76
T\$RPR1	000040	#5-76
T\$SVC	000034	#5-76
T\$T5	000030	#5-76 53-4269
T\$T6	000032	#5-76 53-4270
T.ACT	000031	#5-88
T.CHA	000032	#5-88
T.CSR	000022	#5-88
T.ITM	000001	#5-88
T.NAM	000006	60-47 60-4788
T.QUE	000024	#5-88 5-88
T.QU2	= 000026	#5-88
T.RCVL	000012	39-2873 39-2880 59-4726 59-4733
T.RTY	000030	#5-88
T.ST2	000034	39-2870 59-4722
T.TCBL	000030	*60-4780 60-4782
T.TMR	000000	#5-88
T.TRLG	= 000076	53-4084
T.TRLN	000005	53-4086
T.TRPN	000004	53-4088
T2.FXD	= 002000	39-2870 59-4722
U.CW3	000014	26-1674 26-1776
U.DCB	000000	31-2354 49-3578
U.LEN	= 000042	#7-289 31-2352 49-3577 62-4853
U.UNIT	000006	31-2353
VF\$RCV	= 100000	26-1764
VF\$XMT	= 040000	26-1767
VF.ADD	= ***** GX	26-1527 26-1543
VF.SKA	= ***** GX	12-673
VF.SS	= ***** GX	39-2866 59-4718

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51

.TITLE VENA - VNP ENABLE/DISABLE LINES
.IDENT /V05.00/

.....
COPYRIGHT (C) 1978, 1979, 1980, 1982, 1983, 1985 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

.....
THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

.....
THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

.....
DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

.....
MODULE DESCRIPTION:

VNP - SYSTEM LINE ENABLE/DISABLE ROUTINES

.....
DISTRIBUTED SYSTEMS SOFTWARE ENGINEERING

.....
IDENT HISTORY:

-
1.00 27-FEB-78
VERSION 2.0 RELEASE
.....
2.00 14-DEC-79
DECNET-11M/S V3.0
DECNET-11M-PLUS V1.0
.....
3.00 16-APR-82
DECNET-11M V3.1
DECNET-11M-PLUS V1.1
.....
4.00 07-NOV-83
DECNET-11M V4.0
DECNET-11M-PLUS /2.0
.....
5.00 22-JUL-85
DECnet-11M/S V4.2
DECnet-11M-Plus V3.0
DECnet-Micro/R SX V1.0
.....

\$VENA - ENABLE/DISABLE LINE

524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573

.SBTTL SERBIT - SET/CLE CIRCUIT SERVICE BIT

SERBIT - SET/CLE CIRCUIT SERVICE BIT

INPUTS:

R1 - SLT ADDRESS

R0 - SLT STATION EXTENSION ADDRESS IF MULTIPOINT

OUTPUTS:

LF.SER - SET/CLE

SF.SER - SET/CLE IF MULTIPOINT

```

SERBIT: BIT      #VF.SED,$VFLAG ; SERVICE DISABLED ?
BEQ          10$ ; IF NOT - BRANCH (SERV ENABLE)

BIS          #LF.SER,L.FLG(R1) ; SET SERVICE DISABLED
L.NSTA(R1)   ; MULTIPOINT ?
BEQ          90$ ; IF NO - BRANCH
BISB         #SF.SER,S.FLG(R0) ; SET SERV DISABLED
BR           90$

MOV          L.NSTA(R1),R2 ; GET NUMBER OF TRIBS
BNE          20$ ; IF MULTIPOINT - BRANCH
BIC          #LF.SER,L.FLG(R1) ; CLE SERVICE DISABLE BIT
BR           90$ ; CONTINUE

BICB         #SF.SER,S.FLG(R0) ; CLE SERVICE BIT
MOV          $SLTA,R1 ; GET SLT ADDRESS
ADD          #L.MPF,R1 ; POINT AT STATION TABLE
MOV          #LF.SER,-(SP) ; ASSUME ALL TRIB'S SERVICE ENABLED

BITB         #SF.SER,S.FLG(R1) ; IS TRIB SERV DIABLED ?
BEQ          40$ ; IF NO - BRANCH
CLR          (SP) ; CLEAR LF.SER
BR           50$ ; CONTINUE

ADD          #S.LEN,R1 ; POINT TO NEXT STATION
DEC          R2 ; ONE LESS STATION
BGE          30$ ; IF MORE - BRANCH

MOV          $SLTA,R1 ; GET SLT ADDRESS
BIC          (SP)+,L.FLG(R1) ; CLE SERVICE BIT IF ALL TRIBS
; HAVE SERV ENABLED

90$: RETURN

```

```

1051                                     .SBTTL ENALIN - ENABLE A LINE
1052
1053                                     *** - ENALIN - ENABLE LINE
1054
1055                                     INPUTS:
1056                                     R0 = SLT STATION ADDRESS IF MULTIPOINT LINE
1057                                     R1 = SLT ADDRESS
1058
1059                                     OUTPUTS:
1060                                     C-BIT = SUCCESS/FAILURE
1061
1062                                     -
1063
1064 003134                                     ENALIN:
1065 003134 032767 000000G 000000G      BIT    #VF.DLX,$VFLAG ; IS THIS DLX ONLY SYSTEM ?
1066 003142 001420                                     BEQ    10$ ; IF EQ - NO
1067 003144 052761 000000G 000000G      BIS    #LF.ACT,L.FLG(R1) ; SET LINE'S ACTIVE BIT
1068 003152 112761 000000G 000000G      MOVB   #LN.ON,L.NMST(R1) ; ...
1069 003160 105761 000000G                                     TSTB   L.NSTA(R1) ; MULTIPOINT LINE?
1070 003164 001420                                     BEQ    20$ ; BR IF NO
1071 003166 152760 000000G 000000G      BISB   #SF.ACT,S.FLG(R0) ; MARK TRIB AS ACTIVE
1072 003174 112761 000000G 000000G      MOVB   #LN.ON,S.NMST(R1) ; ...
1073 003202 000411                                     BR     20$
1074
1075 003204 052761 000000G 000000G 10$:   BIS    #LF.ENA,L.FLG(R1) ; SET LINE'S ACTIVE BIT
1076 003212 105761 000000G                                     TSTB   L.NSTA(R1) ; MULTIPOINT LINE?
1077 003216 001403                                     BEQ    20$ ; IF EQ, NOT MULTI-POINT LINE
1078
1079                                     ; MULTI-POINT LINE
1080
1081 003220 152760 000000G 000000G      BISB   #SF.ENA,S.FLG(R0) ; MARK TRIBUTARY AS ACTIVE
1082 003226                                     20$:   RETURN

```



```

1627                                     .SBTTL  FIND PLACE FOR A REMOTE NODE BLOCK
1628                                     ;+
1629                                     ;*** - FINREM - FIND A PLACE FOR A REMOTE NODE BLOCK
1630                                     ;
1631                                     ;INPUTS:
1632                                     ;R1 = NODE ADDRESS OR CHANNEL TO BE INSERTED
1633                                     ;
1634                                     ;OUTPUTS:
1635                                     ;R0 = PRESERVED
1636                                     ;R1 = UNMAPPED ADDRESS OF PREVIOUS REMOTE BLOCK
1637                                     ;R2 = DESTROYED
1638                                     ;-
1639
1640 005500                                     FINREM:
1641 005500 011267 000000G                     MOV    (R2), $BFR                ; GET FIRST IN LIST
1642 005504 010146                                     MOV    R1, -(SP)                ; SAVE NODE ADDRESS ON STACK
1643 005506 010267 000164'                     MOV    R2, CURUNM                ; SAVE UNMAPPED ADDRESS OF CURRENT
1644 005512 016767 000164' 000166' 10$:      MOV    CURUNM, PRVUNM            ; SAVE UNMAPPED ADDRESS OF PREVIOUS
1645 005520 016767 000000G 000164'           MOV    $BFR, CURUNM          ; GET NEXT REMOTE IN LIST
1646 005526 001417                                     BEQ     30$                     ; IF END OF LIST - BRANCH
1647
1648 005530                                     15$:    CEACC$  CURUNM, R2          ; CONVERT TO MAPPED ADDRESS
1649 005542                                     20$:    GETAD   R2, #R.LEN        ; READ REMOTE IN INTERNAL BUFFER
1650 005560 021667 000010G                     CMP     (SP), $BFR+R.ADD        ; DOES REMOTE BLOCK GO HERE ?
1651 005564 101352                                     BHI     10$                     ; IF HI, NO
1652 005566 016701 000166'                     30$:    MOV     PRVUNM, R1        ; RETRIEVE UNMAPPED ADDRESS OF PREVIOUS
1653 005572 005726                                     TST     (SP)+                  ; CLEAN UP THE STACK
1654 005574                                     RETURN
1655

```

```

2130 007702 001521 BEQ 30$ : IF EQ, NO - MUST BE DISMOUNT
2131 007704 CALL $ALL15 : GET I/O PACKET
2132 007710 103565 BCS 109$ : NODE POOL EMPTY
2133 007712 GETRV R5, TLGTH : REREAD TCB
2134 007730 005767 000012G TST T, RCVL+$BFR : ANY PACKETS QUEUED ?
2135 007734 001424 BEQ 17$ : BR IF NO
2136 007736 GETRV T, RCVL+$BFR, #36. : READ IN PACKET
2137 007756 032767 000400 000006G BIT #LS.CX5, LR.STS+6+$BFR : IS SET EXE STATE ON ALREADY QUEUED?
2138 007764 001145 BNE 110$ : BR IF YES - NOTHING TO DO
2139 007766 052767 000400 000006G BIS #LS.CX5, LR.STS+6+$BFR : ELSE SET SET EXE STATE ON ALSO
2140 007774 PUTRC : REWRITE PACKET
2141 010000 CAL $DEA16 : DEALLOCATE PACKET
2142 010004 000537 BR 120$ : AND CONTINUE
2143 010006 010067 000012G 17$: MOV RO, T.RCVL+$BFR : POINT LISHEAD AT FIRST PACKET
2144 010012 010067 000014G MOV RO, T.RCVL+2+$BFR :
2145 010016 PUTRC : REWRITE NTINIT TCB ADDRESS
2146 :
2147 010022 005067 000000G CLR $BFR : CLEAR LINK WORD IN PACKET
2148 010026 012767 131574 000002G MOV #*R..., $BFR+2 : LOAD RAD50 NAME
2149 010034 012767 105700 000004G MOV #*RVNP, $BFR+4 : *-VNP
2150 010042 012767 000401 000006G MOV #LS.CEX!LS.CX5, LR.STS+6+$BFR : ' SET X25-S ON' -> 'SET CEX'
2151 010050 042767 020000 000006G BIC #LS.UNF, LR.STS+6+$BFR : ASSUME NTINIT IS TO REMAIN FIXED
2152 010056 032767 000000G 000000G BIT #VF.FIX, $VFLAG : IS NTINIT TO REMAIN FIXED ?
2153 010064 001003 BNE 20$ : YES
2154 010066 052767 020000 000006G BIS #LS.UNF, LR.STS+6+$BFR : SET FLAG THAT UNFIXES NTINIT
2155 010074 016767 000000G 000040G 20$: MOV .COO, $BFR+32. : ERRORS GO TO COO:
2156 010102 012767 000401 000042G MOV #401, $BFR+34. : [1,1]
2157 010110 PUTRC RO, R1 : REWRITE NODE TO POOL
2158 010124 012704 000001 MOV #TICKS, R4 : NUMBER OF TICKS
2159 010130 CALL RUN : RUN NTINIT
2160 010134 103413 BCS 104$ : NO NODES IN POOL
2161 010136 SWTCHO : POINT BACK AT DEFAULT BUFFER
2162 010144 RETURN :
2163 :
2164 : DISMOUNT REQUEST
2165 :
2166 010146 30$: CALL X5DSMT : DISMOUNT THE ACP
2167 :
2168 010152 100$: RETURN :
2169 :
2170 010154 103$: ERRPT$ ERR45, $EROUT : ACP NOT IN SYSTEM
2171 010164 104$: ERRPT$ ERR9, $EROUT, FATAL : POOL EMPTY
2172 010200 105$: ERRPT$ ERR46, $EROUT : ACP MUST BE FIXED IN FOR RSX11S SYSTEM
2173 010210 010100 107$: MOV R1, RO : GET OTHER NTINIT NAME
2174 010212 CALL TSKSCH : IS IT INSTALLED ?
2175 010216 103406 BCS 1077$ : NO
2176 010220 ERRPT$ ERR15, $EROUT, FATAL : WRONG NTINIT INSTALLED
2177 010234 1077$: ERRPT$ ERR12, $EROUT, FATAL : NTINIT NOT INSTALLED
2178 010250 108$: ERRPT$ ERR14, $EROUT, FATAL : ACP NOT FIXED
2179 010264 109$: ERRPT$ ERR16, $EROUT, FATAL : RSX POOL EMPTY
2180 010300 110$: CALL $DEA16 : DEALLOCATE NODE
2181 010304 120$: RETURN :

```

```

2715          .SBTTL CKMON - CHECK MONITOR SINK NAME
2716          ;+
2717          CKMON - CHECK MONITOR SINK NAME
2718          :
2719          INPUTS:
2720          RQB+LQ.SNM - ASCII SINK NAME
2721          :
2722          OUTPUTS:
2723          CARRY - SUCCESS/FAILURE
2724          :
2725          :-
2726
2727
2728 012676 010546          CKMON: MOV      R5, -(SP)          ; SAVE R5
2729 012700 012705 000154' MOV      #MONNAM, R5      ; POINT TO MONITOR NAME STORAGE
2730 012704 012701 000006 MOV      #MXMON, R1      ; GET MAXIMUM LENGTH OF MONITOR NAME
2731 012710          CALL    GETLEN      ; GET LENGTH OF MONITOR NAME STRING
2732 012714 103424          BCS      40$      ; BR IF INVALID NAME
2733 012716 010003          MOV      R0, R3      ; POINT TO END OF STRING
2734 012720 005303          DEC      R3          ;
2735 012722 160400          SUB      R4, R0      ; POINT TO START OF SINK NODE STRING
2736 012724 012701 000001 10$: MOV      #1, R1      ; PERIODS ARE VALID
2737 012730          CALL    $CAT5      ; CONVERT TO RAD50
2738 012734 103005          BCC      20$      ; BR IF SUCCESS
2739 012736 020427 000003          CMP      R4, #3      ; 3 OR MORE CHARACTERS LEFT?
2740 012742 103010          BHIS     30$      ; BR IF YES - ERROR
2741 012744 020003          CMP      R0, R3      ; AT END OF STRING?
2742 012746 103406          BLO      30$      ; BR IF NO - ERROR
2743 012750 010125          20$: MOV      R1, (R5)+      ; STORE CONVERTED VALUE
2744 012752 162704 000003          SUB      #3, R4      ; SUBTRACT 3 CHARACTERS FROM COUNT
2745 012756 003362          BGT      10$      ; BR IF MORE TO CONVERT
2746 012760 000241          CLC          ; ELSE INDICATE SUCCESS
2747 012762 000401          BR        40$      ; AND FINISH UP
2748 012764 000261          30$: SEC          ; INDICATE FAILURE
2749 012766 012605          40$: MOV      (SP)+, R5      ; RESTORE R5
2750 012770          RETURN

```

```

3230                                     .SBTTL CHNAM - CHANGE LOGGING SINK NAME
3231
3232                                     ;+
3233                                     CHNAM - CHANGE LOGGING SINK NAME
3234
3235                                     INPUTS:
3236                                     RQB - REQUEST BLOCK FOR SET/CLEAR LOGGING COMMAND
3237
3238                                     OUTPUTS:
3239                                     THE SPECIFIED LOGGING SINK NAME IS CHANGED
3240
3241                                     :-
3242
3243
3244 014666 032767 000004 000000G CHNAM: BIT #QF$KND,$QDUAL ; IS THIS FOR KNOWN LOGGING?
3245 014674 001412 BEQ 20$ ; BR IF NO
3246 014676 012702 000020' MOV #DEFFIL,R2 ; POINT TO DEFAULTS
3247 014702 012703 000006G MOV #SLGFNB+6,R3 ; POINT TO CEX DATABASE FOR LOGGING
3248
3249 014706 012701 000030 MOV #DF,LEN,R1 ; GET NUMBER OF WORDS TO MOVE
3250 014712 012223 10$: MOV (R2)+,(R3)+ ; STORE DEFAULTS
3251 014714 005301 DEC R1 ; MORE TO MOVE?
3252 014716 003375 BGT 10$ ; BR IF YES
3253 014720 000472 BR 70$ ; EXIT WHEN ALL DEFAULTS ARE SET UP
3254
3255 014722 032767 000400 000000G 20$: BIT #OP$CON,$O$PTON ; IS THIS FOR LOGGING CONSOLE?
3256 014730 001421 BEQ 30$ ; BR IF NO
3257 014732 032767 000000G 000000G BIT #CM$SET,$CMAND ; IS THIS A SET OPERATION?
3258 014740 001006 BNE 25$ ; BR IF YES
3259 014742 016767 000044' 000160' MOV DEFCON+0,CONNAM+0 ; STORE DEFAULT CONSOLE NAME
3260 014750 016767 000046' 000162' MOV DEFCON+2,CONNAM+2 ; ...
3261
3262 014756 016767 000160' 000000G 25$: MOV CONNAM+0,$LGCON+0 ; STORE CONSOLE NAME
3263 014764 016767 000162' 000002G MOV CONNAM+2,$LGCON+2 ; ...
3264 014772 000445 BR 70$ ; AND EXIT
3265
3266 014774 032767 000100 000000G 30$: BIT #OP$MON,$O$PTON ; IS THIS FOR LOGGING MONITOR?
3267 015002 001421 BEQ 40$ ; BR IF NO
3268 015004 032767 000000G 000000G BIT #CM$SET,$CMAND ; IS THIS A SET OPERATION?
3269 015012 001006 BNE 35$ ; BR IF YES
3270 015014 016767 000040' 000154' MOV DEFMON+0,MONNAM+0 ; STORE DEFAULT CONSOLE NAME
3271 015022 016767 000042' 000156' MOV DEFMON+2,MONNAM+2 ; ...
3272
3273 015030 016767 000154' 000000G 35$: MOV MONNAM+0,$LGMON+0 ; STORE MONITOR NAME
3274 015036 016767 000156' 000002G MOV MONNAM+2,$LGMON+2 ; ...
3275 015044 000420 BR 70$ ; AND EXIT
3276
3277 015046 032767 000000G 000000G 40$: BIT #CM$SET,$CMAND ; IS THIS A SET OPERATION?
3278 015054 001003 BNE 45$ ; BR IF YES
3279 015056 012702 000020' MOV #DEFFIL,R2 ; POINT TO DEFAULT FILE NAME
3280 015062 000402 BR 47$ ; CONTINUE
3281
3282 015064 012702 000134' 45$: MOV #FILNAM,R2 ; POINT TO SPECIFIED FILE NAME
3283 015070 012703 000006G 47$: MOV #SLGFNB+6,R3 ; POINT TO FILE NAME IN CEX COMMON
3284 015074 012701 000020 MOV #FILLER,R1 ; GET LENGTH OF FILE NAME
3285 015100 012223 50$: MOV (R2)+,(R3)+ ; STORE FILE NAME
3286 015102 005301 DEC R1 ; MORE TO MOVE?

```

```

1587 010342 042701 0000000      BIC      #^CARAMSK,R1      ; GET AREA NUMBER
1588
1589      000012      .REPT      10.
1590      .ASR      R1      ; CONVERT TO AREA NUMBER
1591      .ENDM
1592 010372 010167 171160      MOV      R1,TEMP7      ; STORE AREA FOR OUTPUT
1593
1594 010376      ERRPT$      NMS2A,MSGOUT      ; PRINT EXECUTOR AREA ADDRESS
1595
1596      ;
1597      ; DISPLAY STATE AND IDENTIFICATION
1598
1599 010406      250$:      CALL      GSTID      ; GET STATE AND IDENTIFICATION
1600 010412      ERRPT$      NMSG3,MSGOUT      ; DISPLAY STATE AND IDENTIFICATION
1601
1602      ;
1603      ; DISPLAY HOST AND MAXIMUM LINKS
1604 010422 005067 170434      260$:      CLR      TEMP2      ; HOST NODE NAME
1605 010426      GETRV      $DECP1,#D$END      ; READ DECNET HOME BLOCK
1606 010446 016767 000022G 170526      MOV      $BFR+D$HOST,TEMP3      ; STORE HOST ADDRESS
1607 010454 026767 000014G 000022G      CMP      $BFR+D$LNUM,$BFR+D$HOST      ; HOST NODE = EXECUTOR NODE?
1608 010462 001005      BNE      270$      ; BR IF NO
1609 010464 012703 000000G      MOV      #BFR,R3      ; GET NAME
1610 010470 062703 000006      ADD      #D$LNAM,R3
1611 010474 000410      BR      290$      ; DISPLAY IT
1612 010476
1613 010476      270$:      CALL      NXTBLK      ; GET NEXT REMOTE NAME BLOCK
1614 010502 103420      BCS      310$      ; BR IF $HOST NOT FOUND IN REMOTE LIST
1615 010504 021663 000010      CMP      (SP),R.ADD(R3)      ; IS THIS THE HOST NODE?
1616 010510 001372      BNE      280$      ; BR IF NO
1617 010512 062703 000002      ADD      #R.NAM,R3      ; POINT TO NODE NAME
1618 010516 012702 001062      290$:      MOV      #TEMP2,R2      ; STORAGE FOR HOST NODE NAME
1619 000003      .REPT      3
1620      MOV      (R3)+,(R2)+      ; STORE THE NAME
1621      .ENDR
1622 010530 122742 000040      300$:      CMPB      $SPACE,-(R2)      ; DELETE TRAILING BLANKS
1623 010534 001775      BEQ      300$
1624 010536 112762 000000 000001      MOVB      #0,1(R2)      ; CREATE ASCII STRING
1625 010544 005067 170264      310$:      CLR      TEMP1      ; STORAGE FOR MAX ACTIVE LINKS
1626 010550 012700 017704      MOV      #^RECL,R0      ; GET RAD50 PROCESS NAME
1627 010554      CALL      FNDPDV      ; GET ECL'S PDV ADDRESS
1628 010560 103414      BCS      320$      ; BR IF ECL NOT FOUND
1629 010562 010005      MOV      R0,R5      ; SAVE IT
1630 010564      GETRV      Z.DAT(R0),#N$ELEN      ; GET ECL'S DATA BASE
1631 010604 016767 000036G 170222      MOV      N$LVC+$BFR,TEMP1      ; STORE MAXIMUM ACTIVE LINKS
1632
1633 010612 016701 170364      320$:      MOV      TEMP3,R1      ; GET EXECUTOR ADDRESS
1634 010616 042767 000000G 170356      BIC      #ARAMSK,TEMP3      ; STRIP AREA BITS
1635 010624 042701 000000C      BIC      #^CARAMSK,R1      ; GET AREA NUMBER
1636 010630 000241      CLC      ; DO NOT ROR C-BIT
1637
1638      000012      .REPT      10.
1639      ROR      R1      ; POSITION AREA NUMBER
1640      .ENDM
1641
1642 010656 010167 170674      MOV      R1,TEMP7      ; STORE AREA NUMBER
1643 0662      ERRPT$      NMS5A,MSGOUT      ; PRINT AREA ADDRESS

```

```

2243 .SBTTL $DMOD - SHOW MODULE INFORMATION
2244
2245
2246
2247 $DMOD - SHOW MODULE INFORMATION
2248
2249 INPUTS:
2250 $OPTON - OPTIONS WORD
2251 RQB - ADDRESS OF REQUEST BLOCK
2252
2253 OUTPUTS:
2254 SHOW INFORMATION DISPLAYED ON USER'S TERMINAL
2255
2256
2257 $DMOD:: VNPDBG DMOD ; VNP debugging break point ;[TD001]
2258 CALL CKTYPE ; GET COMMAND TYPE ;[TD001]
2259 BCC 10$ ; BR IF SUCCESS ;**-1
2260 ERRPT$ SYNERR,$EROUT ; ELSE SYNTAX ERROR
2261 10$: MOV #MODTBL,R0 ; GET ADDRESS OF MODULE TABLE
2262 20$: MOV (R0)+,R1 ; GET OPTIONS FLAG
2263 BEQ 40$ ; BR IF END OF TABLE
2264 MOV (R0)+,R2 ; GET ADDRESS OF ROUTINE TO CALL
2265 MOV (R0)+,R3 ; GET ADDRESS OF ASCII ID STRING
2266 BIT $OPTON,R1 ; IS THIS THE SPECIFIED OPTION?
2267 BEQ 20$ ; BR IF NO
2268 MOV #TEMP2,R0 ; POINT TO TEMPORARY STORAGE
2269 30$: MOVB (R3)+,(R0)+ ; STORE MODULE ID
2270 BNE 30$ ;
2271 SAVRG <R2> ; SAVE R2
2272 ERRPT$ MOMS1,MSGOUT ; DISPLAY HEADER MESSAGE
2273 RESRG <R2> ; RESTORE R2
2274 CLRB FLAGS ; INITIALIZE FLAGS WORD
2275 CALL (R2) ; CALL ROUTINE TO PROCESS SHOW
2276 40$: RETURN
2277
2278 ; DISPLAY MESSAGES
2279
2280 ERMSG$ <%N% %A %A as of %Y%N>
2281 ERBLK$ MOMS1,<MSPEC,TEMP2,TEMP1>
2282
2283 ;
2284
2285 ; MODULE TABLE
2286 MODTBL: .WORD OP$XAC,DMXAC,MX25AC ; MODULE X25-ACCESS
2287 .WORD OP$XPR,DMXPR,MX25PR ; MODULE X25-PROTOCOL
2288 .WORD OP$XSS,DMXSS,MX25SV ; MODULE X25-SERVER
2289 .WORD OP$X9S,DMX9S,MX29SV ; MODULE X29-SERVER
2290 .WORD 0

```

```

2782 021506      70$:  ERRPT$  GRMS3,MSGOUT      ; GROUP NOT IN SYSTEM
2783 021516      80$:  RETURN
2784
2785      :  MESSAGES
2786      :
2787 021520      ERMSG$ < Group = %A>
2788 021534      ERBLK$ GRMS1,<TEMP1>
2789
2790 021542      ERMSG$ < DTE = %A, Number = %F %A%N>
2791 021600      ERBLK$ GRMS2,<TEMP1,TEMP4,TEMP2>
2792
2793 021612      ERMSG$ < Closed User Group not in system%N>
2794 021657      ERBLK$ GRMS3
  
```

3337
 3338
 3339
 3340
 3341
 3342
 3343
 3344
 3345
 3346
 3347
 3348
 3349
 3350
 3351
 3352
 3353
 3354
 3355
 3356
 3357
 3358
 3359
 3360
 3361
 3362
 3363
 3364
 3365
 3366
 3367
 3368
 3369

025006 000261
 025010 011303
 025012 001415
 025014
 025024
 025040 012703 000000G
 025044 000241
 025046

```

      .SBTTL  NXTBLK - READ IN NEXT BLOCK

      +
      :
      : NXTBLK - READ IN NEXT BLOCK
      :
      : INPUTS:
      :   R3 = ADDRESS OF BUFFER CONTAINING BLOCK - FIRST WORD IS POINTER
      :   TO NEXT BLOCK
      :   R5 = LENGTH OF BLOCK
      :
      : OUTPUTS:
      :   CARRY CLEAR:
      :   R3 = ADDRESS OF BUFFER CONTAINING NEXT BLOCK
      :
      :   CARRY SET:
      :   END OF LIST
      :
      : ALL REGISTERS ARE PRESERVED
      :
      : -
      :
      : .ENABL  LSB

      NXTBLK: SEC                                ; ASSUME END OF LIST
              MOV      (R3),R3                    ; GET NEXT ENTRY
              BEQ      30$                          ; BR IF END OF LIST
              CEACCS$ R3                          ; CONVERT TO MAPPED ADDRESS
      20$:    GETAD   R3,R5                        ; READ IN BLOCK
              MOV     #BFR,R3                      ; GET ADDRESS OF BLOCK
              CLC                                     ; BE SURE CARRY IS CLEAR
      30$:    RETURN

      .DSABL  LSB
  
```


3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3910
3911
3912
3913

027630 005704
027632 001004
027634
027644 012702 001034
027650 016322 000002
027654 016322 000004
027660 016322 000006
027664 122742 000040
027670 001775
027672 112762 000000 000001
027700 012700 114224
027704
027710 016301 000010
027714 006301
027716 060100
027720 016046 000000G
027724 111601
027726 000316
027730 112602
027732 006301
027734 066701 000000G
027740 011101
027742 010100
027744 116101 000002
027750 066701 000000G
027754 011101

```
.SBTTL DISLOO - DISPLAY LOOPBACK NODE INFORMATION

;+
DISLOO - DISPLAY LOOPBACK NODE INFORMATION
INPUTS:
R3 = ADDRESS OF REMOTE NODE BLOCK
R4 = 0 IF HEADER TO BE PRINTED WITH LOOPBACK NODE INFORMATION
    ELSE, ONLY LOOPBACK NODE INFORMATION TO BE PRINTED
OUTPUTS:
LOOPBACK NODE INFORMATION IS DISPLAYED ON USER'S TERMINAL
R0,R1,R2 DESTROYED
-

DISLOO: TST     R4                ; DISPLAY HEADER?
        BNE     10$              ; BR IF NO
        ERRPT$  NLMSG1,MSGOUT    ; DISPLAY HEADER MESSAGE

; STORE NODE NAME
10$:    MOV     #TEMP1,R2        ; STORAGE FOR NODE NAME
        MOV     R.NAM(R3),(R2)+  ; STORE NODE NAME
        MOV     R.NAM+2(R3),(R2)+; ...
        MOV     R.NAM+4(R3),(R2)+; ...
20$:    CMPB    #SPACE,-(R2)     ; DELETE TRAILING BLANKS
        BEQ     20$              ; ...
        MOVB    #0,1(R2)        ; CREATE ASCIZ STRING

; CREATE LINE-ID - DEVICE NAME
        MOV     #*RXPT,R0       ; GET XPT NAME
        CALL    FNDPDV          ; GET XPT'S PDV ADDRESS
        MOV     R.ADD(R3),R1     ; GET CHANNEL NUMBER
        ASL     R1              ; CONVERT CHANNEL NUMBER TO WORD INDEX
        ADD     R1,R0           ; POINT AT CHANNEL IN PDV
        MOV     Z.MAP(R0),-(SP)  ; GET SLN AND TRIBUTARY
        MOVB    (SP),R1         ; GET SLN
        SWAB    (SP),R1         ; GET TRIBUTARY IN LOW BYTE
        MOVB    (SP)+,R2        ; GET TRIBUTARY
        ASL     R1              ; CONVERT SLN TO WORD INDEX
        ADD     $SLTMA,R1       ; GET SLT TABLE ADDRESS
        MOV     (R1),R1         ; GET SLT ADDRESS
        MOV     R1,R0           ; SAVE SLT ADDRESS
        MOVB    L.DDM(R1),R1    ; GET DDM PDV INDEX
        ADD     $PDVTA,R1       ; POINT AT PDV ADDRESS
        MOV     (R1),R1         ; POINT AT DDM PDV
        CALL    FMTLIN          ; FORMAT THE LINE-ID (RETURNED IN TEMP3)
        ERRPT$  NLMSG2,MSGOUT    ; DISPLAY LOOPBACK NODE INFORMATION
        RETURN                  ; FINISHED

.ENABL  LC
```

```

DISLIN - DISPLAY LINE INFORMATION

4489 033522 106200 ASRB R0 ; ...
4490 033524 106200 ASRB R0 ; ...
4491 033526 110067 145450 MOVB R0,TEMP3 ; GET PRIORITY
4492 033532 ERRPT$ LMSG4,MSGOUT ; DISPLAY CSR, VECTOR, PRIORITY
4493 033542 MOV 12(SP),R0 ; RESTORE R0
4494 033546 MOV 10(SP),R1 ; RESTORE R1
4495
4496 ; GET LINE STATE
4497
4498 033552 012705 001034' 80$: MOV #TEMP1,R5 ; STORAGE FOR LINE STATE
4499 033556 012704 000246' MOV #PCLEAR,R4 ; ASSUME IT IS CLEARED
4500 033562 032760 040000 000000 BIT #LF.RDY,L.FLG(R0) ; IS LINE CLEARED?
4501 033570 001425 BEQ 110$ BR IF YES
4502 033572 012704 000077' MOV #OFF,R4 ; ASSUME OFF
4503 033576 032760 000400 000000 BIT #LF.BRO,L.FLG(R0) ; ETHERNET DEVICE
4504 033604 001003 BNE 85$ IF YES - BRANCH
4505 033606 105760 000014 TSTB L,NSTA(R0) ; MULTIPOINT LINE?
4506 033612 001005 BNE 90$ BR IF YES
4507 033614 032760 002000 000000 85$: BIT #LF.ENA,L.FLG(R0) ; IS STATE OFF?
4508 033622 001410 BEQ 110$ BR IF YES
4509 033624 000405 BR 100$ ELSE STATE IS ON
4510 033626 011603 90$: MOV (SP),R3 ; GET MULTIPOINT STATION ADDRESS
4511 033630 132763 BITB #SF.ENA,S.FLG(R3) ; IS STATE OFF?
4512 033636 001402 BEQ 110$ BR IF YES
4513 033640 012704 000064' 100$: MOV #ON,R4 ; ELSE IT IS OFF
4514 033644 112425 110$: MOVB (R4)+,(R5)+ ; STORE THE STATE
4515 033646 001376 BNE 110$
4516
4517 033650 ERRPT$ LMSG6,MSGOUT ; DISPLAY STATE
4518
4519 033660 ERRPT$ LMSG12,MSGOUT ; MAKE OUTPUT LOOK NICE
4520 033670 RESRG <R3,R4,R3,R2,R1,R0>
4521 033704 RETURN
4522
4523 .ENABL LC
4524
4525 ; DISPLAY MESSAGES
4526
4527
4528
4529 033706 ERMSG$ <%N% %A as of %Y%N>
4530 033730 ERBLK$ LMSG1,<TEMP2,TEMP1>
4531
4532 033740 ERMSG$ <%Nline = %A%N>
4533 033755 ERBLK$ LMSG2,<TEMP3>
4534
4535 033762 ERMSG$ < Type = %A, Owner = %A>
4536 034012 ERBLK$ LMSG3,<TEMP2,TEMP3>
4537
4538 034022 ERMSG$ < Controller CSR = %0, Vector = %0, Priority = %B>
4539 034104 ERBLK$ LMSG4,<TEMP1,TEMP2,TEMP3>
4540
4541 034116 ERMSG$ < State = %A>
4542 034133 ERBLK$ LMSG6,<TEMP1>
4543
4544 034140 ERMSG$ < Type = %A>
4545 034154 ERBLK$ LMSG9,<TEMP2>

```

4951
 4952
 4953
 4954
 4955
 4956
 4957
 4958
 4959
 4960
 4961
 4962
 4963
 4964
 4965
 4966
 4967
 4968
 4969
 4970
 4971
 4972
 4973
 4974
 4975
 4976
 4977
 4978
 4979
 4980
 4981
 4982
 4983
 4984

035540 005002
 035542 152102
 035544 010246
 035546 006216
 035550 006216
 035552 006216
 035554 006216
 035556 062716
 035562 113623
 035564 042702
 035570 062702
 035574 111223
 035576 005300
 035600 003357
 035602 105023
 035604

001575'
 177760
 001575'

```

.SBTTL  HEXASC - CONVERT HEX TO ASCIZ

+
:
:  HEXASC - CONVERT HEX TO ASCIZ
:
:  INPUTS:
:    R0 - NUMBER OF BYTES IN HEX NUMBER
:    R1 - ADDRESS OF BUFFER CONTAINING HEX DIGITS
:    R3 - ADDRESS OF BUFFER TO STORE CONVERTED HEX DIGIT
:
:  OUTPUTS:
:    R1 - POINTER PAST HEX DIGIT
:    R3 - POINTER TO NEXT AVAILABLE BYTE IN CONVERSION BUFFER
:    R0,R2 DESTROYED
:
:
:  HEXASC: CLR      R2                ; GET NEXT TWO HEX DIGITS
:           BISB    (R1)+,R2          ; ...
:           MOV     R2,-(SP)          ; ...
:           ASR     (SP)              ; ISOLATE HIGH FOUR BITS
:           ASR     (SP)              ; ...
:           ASR     (SP)              ; ...
:           ASR     (SP)              ; ...
:
:           ADD     #CVTHEX,(SP)      ; INDEX INTO HEX TABLE
:           MOVB    @((SP)+,(R3)+    ; STORE ASCII HEX DIGIT
:           BIC     #177760,R2        ; ISOLATE LOW FOUR BITS
:           ADD     #CVTHEX,R2        ; INDEX INTO HEX CONVERSION TABLE
:           MOVB    (R2),(R3)+        ; STORE ASCII HEX DIGIT
:           DEC     R0                ; MORE TO CONVERT?
:           BGT     HEXASC            ; BR IF YES
:           CLRB    (R3)+             ; CREATE ASCIZ STRING
:
:  RETURN

```

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 5 M 8

SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
		*29-2274 *31-2332 31-2364 *33-2441 33-2480 35-2667 *36-2721 37-2776 *38-2832
		40-3059 *41-3149
FMT CIR	034334 R	53-3959 #57-4621
FMT LIN	034166 R	35-2663 52-3907 55-4404 #56-4569
FNDADL	007276 R	18-1035 #25-1396
FNDBLK	024620 R	27-1955 27-2014 27-2097 32-2410 33-2439 38-2830 41-3147 #44-3275
FNDPDV	035140 R	12-664 26-1574 26-1577 26-1627 26-1664 26-1759 34-2527 39-2850 52-3892
		53-4006 53-4035 53-4066 #61-4813
FNDUCB	035212 R	26-1673 26-1775 #62-4847
F.ADD	000016	#5-84 22-1293 22-1309
F.CEV	000030	#5-84 20-1193 20-1194
F.CIR	000024	#5-84
F.CLS	000002	#5-84 12-727 19-1101 19-1108 19-1127
F.EVT	000004	#5-84 19-1088 19-1089 19-1090 19-1091 19-1103 19-1104 19-1105 19-1106
F.FLG	000014	#5-84 12-732 17-979 17-986 19-1093
F.FND	= 000004	#7-325 31-2332 31-2364 33-2441 33-2480 35-2667 36-2721 37-2776 38-2832
		40-3059 41-3149
F.LEN	000016	#5-84 12-729
F.LIN	000020	#5-84 21-1225 21-1261
F.LNK	000000	#5-84
F.MOD	000022	#5-84
F.PREV	= 000002	#7-324 19-1144 19-1148 19-1155 23-1344 23-1350
F.REM	000026	#5-84 17-984 17-993 17-995
F.RSIZ	= ***** GX	*10-461 *18-1023 *22-1298 *22-1308 *25-1397 *25-1408 *26-1439 *26-1484 *26-1524
		*26-1605 *26-1630 *26-1684 *26-1706 *26-1713 *26-1762 *26-1763 *27-1949 *27-2003
		*27-2091 *31-2329 *31-2352 *31-2354 *33-2430 *34-2553 *34-2570 *35-2635 *37-2741
		*37-2767 *39-2880 *39-2894 *40-2926 *42-3168 *42-3180 *44-3280 *45-3364 *49-3577
		*49-3578 *53-4021 *53-4025 *53-4049 *53-4053 *53-4080 *53-4084 *53-4208 *53-4258
		*53-4262 *55-4473 *58-4682 *59-4733 *59-4743 *59-4749 *60-4778 *62-4850 *62-4853
F.SEV	000004	#5-84 20-1197 20-1198 20-1199 20-1200
F.STRT	= 000001	#7-323 19-1138 19-1143
F.URBD	= ***** GX	*10-460 *10-462 *18-1022 *18-1024 *22-1297 *22-1301 *22-1323 *25-1396 *25-1400
		*25-1412 *26-1761 *31-2350 *31-2357 *49-3575 *49-3588
GCLASS	= 001044 R	#7-315 7-316 *19-1101 *19-1102 *19-1110
GEVMSK	= 001046 R	#7-316 *19-1103 *19-1104 *19-1105 *19-1106 20-1189 20-1190 20-1191 20-1192
GF\$BUG	= 000001	37-2758
GRMS1	021536 R	37-2754 #37-2788
GRMS2	021602 R	37-2773 #37-2791
GRMS3	021660 R	37-2782 #37-2794
GRPBIL	000750 R	#6-233 37-2761
GTEXEC	034542 R	26-1460 26-1583 #58-4682
GTSTID	034630 R	26-1475 26-1599 26-1814 #59-4712
G\$CUG	000010	63-4872
G\$DTE	000012	37-2765
G\$FLG	000014	37-2758
G\$GNAM	000016	38-2828
G\$LEN	000016	38-2823 38-2827
G\$NAM	000002	38-2824
G\$NML	000015	38-2829
HEXASC	035540 R	40-3014 40-3017 #65-4968 65-4982
H\$CUG	000010	37-2740
H\$DST	000012	39-2856

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 18 M 9
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
V\$FLG	000000	26-1764 26-1767
V\$LEN	000022	26-1763
WC.CNT	= 000400	#5-99 27-2026 28-2199
WC.EVE	= 004000	#5-99
WC.TRI	= 002000	#5-99 27-2042
WC.UNT	= 001000	#5-99 27-2033 28-2206
XPER1	020062 R	34-2529 #34-2603
XPMS1	020162 R R	34-2574 #34-2608
XPMS2	020272 R R	34-2581 #34-2611
XPMS3	020402 R R	34-2589 #34-2614
XPMS4	020510 R R	34-2597 #34-2617
XSER1	022264 R R	39-2852 #39-2902
X5MS1	022334 R R	39-2896 #39-2907
X9ER1	024442 R R	42-3171 #42-3190
X9MS1	024522 R	42-3183 #42-3195
Y\$DPSZ	000002	34-2571
Y\$DWSZ	000006	34-2573
Y\$LEN	000020	34-2570
Y\$MCLR	000016	34-2580
Y\$MPSZ	000004	34-2576
Y\$MRES	000015	34-2584
Y\$MRST	000014	34-2586
Y\$MWSZ	000037	34-2578
Y\$TCAL	000011	34-2588
Y\$TCLR	000013	34-2592
Y\$TRES	000012	34-2594
Y\$TRST	000010	34-2596
ZF.MUX	= ***** GX	21-1247 56-4571 57-4623
Z\$CTIM	000014	39-2895
Z\$LEN	000026	39-2894
Z.DAT	= ***** GX	26-1630 26-1762 34-2565 39-2889 53-4258
Z.DSP	= ***** GX	34-2569 39-2893
Z.FLG	= ***** GX	21-1247 56-4571 57-4623
Z.LLN	= ***** GX	26-1666
Z.MAP	= ***** GX	52-3896
Z.NAM	= ***** GX	1-606 21-1235 27-2021 28-2176 28-2192 28-2194 28-2222 46-3392 46-3397
		50-3701 53-3980 53-4118 53-4248 53-4423 55-4423 55-4430 55-4446 56-4574
		57-4626 61-4819
Z.PCB	= ***** GX	11-591 50-3709
\$BFR	= ***** GX	10-462 12-681 12-727 18-1024 22-1323 25-1412 26-1440 26-1485 26-1525
		26-1529 26-1531 26-1533 26-1537 26-1606 26-1607 26-1607 26-1609 26-1631
		26-1674 26-1679 26-1685 26-1686 26-1690 26-1692 26-1693 26-1697 26-1698
		26-1707 26-1708 26-1714 26-1719 26-1721 26-1723 26-1725 26-1727 26-1738
		26-1746 26-1748 26-1761 26-1763 26-1764 26-1767 26-1776 26-1776 26-1776
		31-2345 31-2348 31-2352 31-2357 33-2452 33-2457 33-2459 33-2465 33-2468
		33-2470 34-2555 34-2571 34-2573 34-2576 34-2578 34-2580 34-2584 34-2586
		34-2588 34-2592 34-2594 34-2596 35-2638 35-2639 35-2645 35-2652 35-2664
		36-2705 36-2711 36-2714 37-2738 37-2764 37-2765 37-2768 37-2769 37-2772
		39-2870 39-2873 39-2880 39-2883 39-2895 40-2945 40-2952 40-2953
		40-2955 40-2958 40-2960 40-2967 40-2971 40-2980 40-2981 40-2984 40-2996
		40-2998 40-2999 40-3003 40-3004 40-3010 40-3020 40-3030 40-3031 40-3033
		40-3037 40-3039 40-3050 40-3051 40-3051 40-3055 42-3169 42-3179 42-3181

```

53          .SBTTL  MACRO CALLS
54
55          ;
56          ; MACRO CALLS
57          ;
58          ;
59          .MCALL  ERMSG$,ERBLK$,ERRPT$,GETRV$,PUTRC$,TCBDF$,CLKDF$,XPDDB$,ECddb$
60          .MCALL  SWTCHO$,VNPdF$,RNBdF$,OBJdF$,ANBdF$,PUTAD$,GETAD$,NSFdF$
61          .MCALL  UCBDf$,DDCdF$,PCLdF$,FLTdF$,SWTCH1$,CEACC$,EVLdF$,PLBdF$
62          .MCALL  SAVRG$,RESRG$,ASL$,CTRDf$,DMPdF$,ADJdF$,CALLR$,DHBdF$
63
64          .ENABL  LC
65
66          000000  TCBdF$          ; TCB OFFSETS
67          000000  CLKdF$          ; CLOCK QUEUE ENTRY OFFSETS
68          000000  CTRdF$          ; DEFINE COUNTER BLOCK OFFSETS
69          000000  FLTdF$          ; DEFINE FILTER BLOCK OFFSETS
70          000000  ECddb$          ; DEFINE ECL DATA BASE OFFSETS
71          000000  XPddb$          ; DEFINE XPT DATA BASE OFFSETS
72          000000  PLBdF$          ; DEFINE PLB OFFSETS
73          000000  VNPdF$          ; DEFINE VNP OFFSETS
74          000000  RNBdF$          ; DEFINE REMOTE BLOCK OFFSETS
75          000000  OBJdF$          ; DEFINE OBJECT BLOCK OFFSETS
76          000000  ANBdF$          ; DEFINE ALIAS BLOCK OFFSETS
77          000000  NSFdF$          ; DEFINE NSP FEATURES MASK BITS
78          000000  UCBDf$          ; DEFINE UCB OFFSETS
79          000000  EVLdF$          ; DEFINE EVENT CLASSES AND TYPES
80          000000  DMPdF$          ; DEFINE DMP OFFSETS
81          000000  DDCdF$          ; DEFINE DDCMP OFFSETS
82          000000  PCLdF$          ; DEFINE PCL OFFSETS
83          000000  ADJdF$          ; DEFINE ADJACENCY DATA BASE OFFSETS
84          000000  DHBdF$          ; DEFINE DECNET HOME BLOCK OFFSETS
85
86          ;
87          ; LOCAL MACRO CALLS
88          ;
89          ;
90          ;
91          ; SHIFT LEFT
92          ;
93          .MACRO  ASL$  COUNT,REG
94          .IF DF  R$$$EIS
95          ASH    #'COUNT',REG
96          .IFF
97          .REPT  COUNT
98          ASL    REG
99          .ENDR
100         .ENDC
101         .ENDM  ASL$
102
103         .MACRO  ASR$  COUNT,REG
104         .IF DF  R$$$EIS
105         ASH    #-'COUNT',REG
106         .IFF
107         .REPT  COUNT
108         ASR    REG
109         .ENDR

```

```

$VTAD - SET TRIBUTARY STATION ADDRESS

575                                .SBTTL $VTAD - SET TRIBUTARY STATION ADDRESS
576
577                                ;+
578                                ;**-$VTAD-SET TRIBUTARY STATION ADDRESS
579                                ; THIS ROUTINE IS CALLED TO CHANGE THE TRIBUTARY ADDRESS
580                                ; FOR A MULTIPOINT LINE.
581                                ;
582                                ; INPUTS:
583                                ;     $SLTA = SLT ADDRESS
584                                ;
585                                ; OUTPUTS:
586                                ;     C-BIT = SUCCESS/FAILURE
587                                ;
588                                ; -
589
590                                $VTAD::
591                                001230 016701 000000G MOV $SLTA,R1 ; GET SLT ADDRESS
592                                001234 032761 000000G 000000G BIT #LF.MTP,L.FLG(R1) ; IS THIS A MULTI-POINT LINE ?
593                                001242 001432 BEQ 1%1$ ; IF EQ, NO : ERROR
594                                001244 CALL FNDTRI ; FIND THE DCP TRIBUTARY TABLE
595                                001250 103426 BCS 40$ ; IF CS, ERROR DETECTED
596                                001252 001404 BEQ 10$ ; IF EQ, LINE TABLE NOT FOUND
597                                001254 116767 000030G 000034G MOVB R0B+LO.TRI,$BFR+S.STPN ; ELSE, SET THE NEW TRIB ADDRESS
598                                001262 000421 BR 40$ ; AND RETURN
599                                001264 10$: CALL FNDPCL ; MAYBE IT'S A PCL ?
600                                001270 103416 BCS 40$ ; IF CS, ERROR DETECTED
601                                001272 001404 BEQ 20$ ; IF EQ, LINE TABLE NOT FOUND
602                                001274 116767 000030G 000003G MOVB R0B+LO.TRI,$BFR+S.PSA ; ELSE, SET NEW PHYSICAL ADDRESS
603                                001302 000411 BR 40$ ; AND RETURN
604                                001304 20$: CALL FNDDMP ; MAYBE IT'S A DMP?
605                                001310 103406 BCS 40$ ; IF CS, ERROR DETECTED
606                                001312 001412 BEQ 102$ ; IF EQ, LINE TABLE NOT FOUND
607                                001314 116767 000030G 000004G MOVB R0B+LO.TRI,$BFR+T.TRPN ; ELSE SET NEW PHYSICAL ADDRESS
608                                001322 PUTAD ; WRITE BACK THE LINE TABLE
609
610                                40$: RETURN ; BACK TO USER STATE
611
612                                ; ERROR CONDITIONS
613
614                                101$: ERRPT$ ERR31,$EROUT ; LINE IS NOT MULTIPOINT
615                                001330 102$: ERRPT$ ERR32,$EROUT ; INVALID PARAMTER, TRIBUTARY ADDRESS

```

```

1084          .SBTTL DSBLIN - DISABLE A LINE
1085
1086          *** - DSBLIN - DISABLE LINE
1087
1088          INPUTS:
1089              R0 = SLT STATION ADDRESS IF MULTIPOINT LINE
1090              R1 = SLT ADDRESS
1091
1092          OUTPUTS:
1093              C-BIT = SUCCESS/FAILURE
1094
1095          -
1096
1097          DSBLIN:
1098      003230      032767      000000G 000000G      BIT      #VF.DLX,$VFLAG      ; IS THIS A DLX ONLY SYSTEM
1099      003236      001445      000000G      BEQ      50$      ; BR IF NO
1100      003240      116102      000000G      MOVVB   L,NSTA(R1),R2      ; MULTIPOINT LINE?
1101      003244      001007      000000G      BNE     10$      ; IF NE, YES
1102
1103          ;
1104          PT-TO-PT LINE - DLX ONLY
1105      003246      042761      000000G 000000G      BIC     #LF.ACT,L.FLG(R1) ; CLEAR LINE'S ACTIVE BIT
1106      003254      112761      000000G 000000G      MOVVB   #LN.OFF,L.NMST(R1) ; SET MANAGEMENT STATE TO 'OFF'
1107      003262      000471      000000G      BR      100$      ; AND RETURN
1108
1109          ;
1110          MULTI-POINT LINE - DLX ONLY
1111      003264      142760      000000G 000000G      10$: BICB     #SF.ACT,S.FLG(R0) ; CLEAR ACTIVE BIT
1112      003272      112760      000000G 000000G      MOVVB   #LN.OFF,S.NMST(R0) ; SET STATE TO 'OFF'
1113      003300      016701      000000G      MOV      $SLTA,R1      ; GET SLT ADDRESS
1114      003304      062701      000000G      ADD      #L.MPF,R1      ; POINT TO STATION TABLE
1115      003310      012746      000000G      MOV      #LF.ACT,-(SP) ; ASSUME ALL TRIBUTARIES ARE 'OFF'
1116      003314      132761      000000G 000000G      20$: BITB     #SF.ACT,S.FLG(R1) ; IS TRIBUTARY ACTIVE ?
1117      003322      001402      000000G      BEQ      30$      ; IF EQ, NO
1118      003324      005016      000000G      CLR      (SP)      ; ELSE, DON'T CLEAR ACTIVE BIT IN SLT
1119      003326      000404      000000G      BR      40$      ; AND LEAVE
1120      003330      062701      000000G      30$: ADD      #S.LEN,R1      ; GET NEXT SLT TRIB EXTENSION
1121      003334      005302      000000G      DEC      R2      ; ANY MORE TRIBUTARIES ?
1122      003336      002366      000000G      BGE     20$      ; IF YES - BRANCH
1123      003340      016701      000000G      40$: MOV      $SLTA,R1      ; ELSE, GET SLT ADDRESS
1124      003344      042661      000000G      BIC     (SP)+,L.FLG(R1) ; CLEAR ACTIVE BIT IF ALL TRIBS ARE 'OFF'
1125      003350      000436      000000G      BR      100$      ; AND EXIT
1126
1127          ;
1128          POINT-TO-POINT NON-DLX
1129      003352      116102      000000G      50$: MOVVB   L,NSTA(R1),R2      ; GET NUMBER OF TRIBS
1130      003356      001004      000000G      BNE     60$      ; BR IF MULTIPOINT LINE
1131      003360      042761      000000G 000000G      BIC     #LF.ENA,L.FLG(R1) ; TURN CIRCUIT OFF
1132      003366      000427      000000G      BR      100$      ; AND EXIT
1133
1134          ;
1135          MULTI-POINT NON-DLX
1136      003370      142760      000000G 000000G      60$: BICB     #SF.ENA,S.FLG(R0) ; TURN CIRCUIT OFF
1137      003376      016701      000000G      MOV      $SLTA,R1      ; GET SLT ADDRESS
1138      003402      062701      000000G      ADD      #L.MPF,R1      ; POINT TO STATION TABLE
1139      003406      012746      000000G      MOV      #LF.ENA,-(SP) ; ASSUME ALL TRIBUTARIES ARE 'OFF'
1140      003412      132761      000000G 000000G      70$: BITB     #SF.ENA,S.FLG(R1) ; IS TRIBUTARY ACTIVE ?
  
```



```

1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675 005576 005767 000000G
1676 005602 001002
1677 005604 001002
1678 005606
1679
1680 005610 020167 000000G
1681 005614 103773
1682 005616 010167 000000G
1683 005622 016701 000000G
1684 005626 010067 000000G
1685 005632 000241
1686 005634
1687
1688

      .SBTTL  QUICK - FAST REMOTE NODE PROCESSING
      :+
      : QUICK - FAST REMOTE NODE PROCESSING
      :
      : INPUTS:
      : R0      - POINTER TO NEW NODE BLOCK
      : R1      - NODE ADDRESS TO BE INSERTED
      : $LSTAD  - CURRENT HIGHEST NODE ADDRESS
      : $LSTRM  - HIGHEST NODE ADDRESS POINTER
      :
      : OUTPUTS:
      : C SET   - FAST SEARCH FAILURE
      : C CLR   - FAST SEARCH SUCCESS
      : R1      - POINTER TO NEW PREVIOUS BLOCK
      : $LSTAD,$LSTRM - UPDATED
      :-
      :
      : QUICK:  TST      $LSTAD      ; HIGH ADDRESS VALID ?
      :          BNE     10$         ; IF YES - BRANCH
      : 5$:      SEC          ; INDICATE FAILURE
      :          RETURN
      :
      : 10$:     CMP      R1,$LSTAD   ; INSERT AT END OF LIST ?
      :          BLO     5$          ; IF NO - BRANCH
      :          MOV     R1,$LSTAD   ; UPDATE NEW HIGH ADDRESS
      :          MOV     $LSTRM,R1   ; SET NEW PREVIOUS NODE BLOCK POINTER
      :          MOV     R0,$LSTRM   ; UPDATE NEW LAST NODE BLOCK POINTER
      :          CLC          ; INDICATE SUCCESS
      :          RETURN

```

```

2183          .SBTTL $VOBJ - SET/CLEAR OBJECT BLOCK
2184
2185          *** - $VOBJ - SET/CLEAR OBJECTS
2186
2187          INPUTS:
2188          $OPTON = OPTIONS WORD
2189          REQUEST BLOCK CONTAINS DATA
2190
2191          OUTPUTS:
2192          NONE.
2193
2194          -
2195
2196          010306          $VOBJ:: MOV      R0B+LR.TYP,R1      ; GET THE OBJECT TYPE CODE
2197          010306          016701 000002G      BIT      #CM$SET,$CMAND ; IS THIS A SET COMMAND ?
2198          010312          032767 000000G      BNE      20$      ; IF NE, YES
2199          010320          001045
2200
2201          : CLEAR REQUEST
2202
2203          010322          032767 000004 000000G      BIT      #QF$KNO,$QUAL ; IS THIS FOR KNOWN OBJECTS ?
2204          010330          001004      BNE      1$      ; IF NE, YES
2205          010332          005701      TST      R1      ; ELSE, IS THIS OBJECT TYPE ZERO ?
2206          010334          001002      BNE      1$      ; IF NE, NO .. OKAY
2207          010336          000167 000440      JMP      103$      ; ELSE, ERROR
2208          010342          1$: CALL      $FNOBJ      ; TRY TO FIND THE OBJECT
2209          010346          103006      BCC      3$      ; IF CC, FOUND .. OKAY
2210          010350          032767 000004 000000G      BIT      #QF$KNO,$QUAL ; ELSE, WAS THIS FOR KNOWN OBJECTS ?
2211          010356          001121      BNE      49$      ; IF NE, YES .. OKAY
2212          010360          000167 000376      JMP      101$      ; ELSE, ERROR
2213          010364          022701 000000G      CMP      #SOBJHD,R1      ; IS PREVIOUS OBJECT THE LISTHEAD ?
2214          010370          001004      BNE      5$      ; IF NE, NO
2215          010372          016767 000000G 000000G      MOV      $BFR,$OBJHD      ; ELSE, SET NEW NEXT
2216          010400          000404      BR      15$      ; AND CLEAN UP
2217
2218          010402          010146      5$: MOV      R1,-(SP)      ; GET UNMAPPED ADDR OF PREVIOUS BLOCK
2219          010404          010046      MOV      R0,-(SP)      ; GET UNMAPPED ADDRESS OF CURRENT BLOCK
2220          010406          010046      CALL      $XLINK      ; UNLINK FROM LIST
2221          010412          012701 000012      15$: MOV      #Q.LEN,R1      ; GET LENGTH OF OBJECT BLOCK
2222          010416          010046      CALL      $DEPOL      ; DEALLOCATE THE OBJECT BLOCK
2223          010422          032767 000004 000000G      BIT      #QF$KNO,$QUAL ; IS THIS FOR KNOWN OBJECTS?
2224          010430          001344      BNE      1$      ; BR IF YES
2225          010432          000473      BR      49$      ; ELSE FINISHED
2226
2227          : SET REQUEST
2228
2229          010434          20$: CALL      $FNOBJ      ; TRY TO FIND THE OBJECT BLOCK
2230          010440          103471      BCS      50$      ; IF CS, NOT FOUND .. MAKE NEW ONE
2231          010442          032767 000001 000000G      BIT      #OP$COP,$OPTON ; WERE THE NO. OF COPIES GIVEN ?
2232          010450          001415      BEQ      30$      ; IF EQ, NO
2233          010452          116767 000010G 000004G      MOVBS  R0B+LO.COP,$BFR+D.MXC ; ELSE, STORE NO. OF COPIES
2234          010460          152767 000200 000003G      BISB   #OF.SMC,$BFR+D.FLG ; ASSUME MULTIPLE COPIES
2235          010466          032767 000200 000012G      BIT      #QF$SMC,R0B+LO.OFL ; SPAWN MULTIPLE COPIES ?
2236          010474          001003      BNE      30$      ; IF NE, YES .. OKAY
2237          010476          142767 000200 000003G      BICB   #OF.SMC,$BFR+D.FLG ; ELSE, CLEAR BIT
2238          010504          032767 000002 000000G 30$: BIT      #OP$USE,$OPTON ; WAS THE USER REQUEST SPECIFIED ?
2239          010512          001412      BEQ      40$      ; IF EQ, NO

```

```

2752                                     .SBTTL CKCON - CHECK CONSOLE NAME FOR VALIDITY
2753
2754                                     ;*
2755                                     ;
2756                                     ; CKCON - CHECK CONSOLE NAME FOR VALIDITY
2757                                     ;
2758                                     ; INPUTS:
2759                                     ;   RQB+LO.SNM - CONSOLE SINK NAME
2760                                     ;
2761                                     ; OUTPUTS:
2762                                     ;   CARRY - SUCCESS/FAILURE
2763                                     ;
2764                                     ; -
2765
2766 012772 012701 000006      CKCON:  MOV     #MXCON,R1      ; GET MAXIMUM LENGTH OF CONSOLE NAME
2767 012776                  CALL    GETLEN      ; GET LENGTH OF CONSOLE NAME STRING
2768 013002 103432          BCS     20$         ; BR IF ERROR
2769 013004 160400          SUB     R4,R0        ; POINT TO START OF SINK NAME
2770 013006 012704 000160'  MOV     #CONNAM,R4    ; POINT TO CONSOLE NAME STORAGE
2771 013012          CALL    CKMDEV      ; CHECK DEVICE AND UNIT NUMBER
2772 013016 103424          BCS     20$         ; BR IF NOT RSX11M DEVICE
2773 013020 016702 000000C  MOV     D,UCB+$BFR,R2 ; GET UCB ADDRESS
2774 013024          GETRV   R2,#U.CNT+2      ; READ IN UCB
2775 013042 026702 000002G  CMP     U,RED+$BFR,R2 ; HAS THIS DEVICE BEEN REDIRECTED?
2776 013046 001403          BEQ     15$         ; BR IF NO
2777 013050 016702 000002G  MOV     U,RED+$BFR,R2 ; GET NEXT UCB ADDRESS IN LIST
2778 013054 000763          BR      10$        ; CHECK FOR NEXT UCB IN REDIRECT LIST
2779 013056 032767 000001 000010G 15$: BIT     #DV.REC,U.CW1+$BFR ; IS IT A TERMINAL DEVICE
2780 013064 001001          BNE     20$        ; BR IF YES
2781 013066 000261          SEC
2782 013070          20$:    RETURN          ; INDICATE ERROR

```

M 16
VENA - VNP ENABLE/DISABLE LINES MACRO V05.03b Wednesday 11-Sep-85 12:06 Page 58-1
CHNAM - CHANGE LOGGING SINK NAME

3287 015104 003375

BGT 50\$

; BR IF YES

3288
3289 015106

70\$: RETURN

```

1644 010672 000426          BR      360$          ; CONTINUE
1645
1646 010674          330$:  ERRPT$  NMSG5,MSGOUT    ; DISPLAY HOST AND MAX ACTIVE LINKS
1647 010704 000421          BR      360$          ;
1648
1649
1650
1651 010706 000241          350$:  CLC              ; DO NOT ROR C-BIT
1652 000012              .REPT      10.            ;
1653              ROR      R1                    ; POSITION AREA BITS
1654              .ENDM
1655 010734 010167 170616    MOV      R1,TEMP7      ; STORE AREA FOR OUTPUT
1656
1657
1658 010740              ERRPT$  NMS12A,MSGOUT      ; PRINT AREA ADDRESS
1659
1660              ;
1661              ; GET MAXIMUM CIRCUITS
1662
1663 010750 012700 114224    360$:  MOV      #*RXF,RO    ; GET XPT PROCESS NAME
1664 010754              CALL     FNDPD,            ; GET XPT'S PDV ADDRESS
1665 010760 005067 170050    CLR      TEMP1          ; INIT NUMBER OF CIRCUITS
1666 010764 156067 090000G 170042    BISB     Z,LLN(RO),TEMP1 ; GET MAXIMUM CIRCUITS+1
1667 010772 005367 170036    DEC      TEMP1          ; DO NOT INCLUDE CHANNEL 0
1668
1669              ;
1670              ; DETERMINE ROUTING TYPE
1671
1671 010776 012700 000416'   MOV      #ROUTING,RO    ; ASSUME ROUTING NODE
1672 011002 012701 001560'   MOV      #TEMP8,R1      ; STORAGE FOR ROUTING TYPE STRING
1673 011006              CALL     FNDUCB           ; GET UCB FOR NS: DEVICE
1674 011012 032767 000200 000014G BIT      #NF.END,$BFR+U.CW3 ; END NODE?
1675 011020 001402          BEQ      370$          ; BR IF NO
1676 011022 012700 000426'   MOV      #NRROUT,RO    ; NONROUTING NODE
1677 011026 112021          MOVVB   (RO)+,(R1)+      ; STORE ROUTING TYPE STRING
1678 011030 001376          BNE      370$          ;
1679 011032 032767 000200 000014G BIT      #NF.END,$BFR+U.CW3 ; ROUTING NODE?
1680 011040 001075          BNE      400$          ; BR IF NO
1681
1682              ;
1683              ; DISPLAY MAXIMUM ADDRESS, MAXIMUM CIRCUITS, AND MAXIMUM COST
1684
1684 011042              GETRV     $DECPT,$D$END      ; READ DECNET HOME BLOCK
1685 011062 016767 000064G 170112    MOV      $BFR+$D$AMXC,TEMP3 ; STORE MAX COST
1686 011070 016767 000060G 170126    MOV      $BFR+$D$NN,TEMP4   ; STORE MAX ADDRESS
1687 011076
1688 011076          380$:  ERRPT$  NMSG6,MSGOUT    ; DISPLAY MESSAGE
1689
1690 011106 016767 000062G 167720    MOV      $BFR+$D$NA,TEMP1    ; GET MAXIMUM AREA
1691 011114 001412          BEQ      390$          ; IF NONE - BRANCH
1692 011116 016767 000072G 167736    MOV      $BFR+$D$AMXC,TEMP2    ; STORE MAX AREA COST
1693 011124 016767 000074G 170050    MOV      $BFR+$D$AMXH,TEMP3    ; STORE MAX AREA HOPS
1694
1695 011132              ERRPT$  NMSG6A,MSGOUT      ; DISPLAY MULTIPLE AREA CHAR
1696
1697 011142 016767 000054G 170054    390$:  MOV      $BFR+$D$NBRA,TEMP4    ; STORE ROUTER ADJ.
1698 011150 016767 000056G 170400    MOV      $BFR+$D$NBEA,TEMP7    ; STORE ENDNODE ADJ.
1699
1700 011156              ERRPT$  NMSG6B,MSGOUT      ; DISPLAY EXECUTOR ADJACENCIES

```

VDIS - VNP SHOW COMMANDS
DMXAC - SHOW MODULE X25-ACCESS

MACRO V05.03b Monday 15-Jul-85 12:14^{N 2} Page 30

```
2292 .SBTTL DMXAC - SHOW MODULE X25-ACCESS
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305 015724 032767 010000 000000G DMXAC: BIT #OP$KNT,$OPTON ; Check if displaying KNOWN NETWORKs ;[TD001]
2306 015732 001007 000000G BNE 10$ ; Branch if yes ;[TD001]
2307 015734 032767 004000 000000G BIT #OP$NET,$OPTON ; Check if displaying NETWORK xxx ;[TD001]
2308 015742 001003 BNE 10$ ; Branch if yes ;[TD001]
2309
2310 015744 CALL DMXACD ; Display X25-ACCESS destinations ;[TD001]
2311 015750 RETURN ;[TD001]
2312
2313 015752 10$: CALL DMXACN ; Display X25-ACCESS networks ;[TD001]
2314 015756 RETURN ;[TD001]
```

VDIS - VNP SHOW COMMANDS
NXTGRP - Get next group block

MACRO V05.03b Monday 15-Jul-85 12:14 ^{N 3} Page 38

```

2796 .JBTTL NXTGRP - Get next group block ;[TD001]
2797 ;*-1
2798
2799
2800 *** NXTGRP - Get next group block ;[TD001]
2801 ;[TD001]
2802 INPUTS: ;[TD001]
2803 $BFR - Address of next group descriptor ;[TD001]
2804 $OPTDN - Options word ;[TD001]
2805 LD.GRO+RQB - Group name to look for ;[TD001]
2806 ;*-6
2807
2808 OUTPUTS:
2809 CARRY CLEAR: ;[TD001]
2810 $BFR contains next group block ;[TD001]
2811 R0 - Address of group name in block ;[TD001]
2812 R1 - Length of group name in block ;[TD001]
2813 CARRY SET: ;[TD001]
2814 End of closed user group blocks ;[TD001]
2815 Registers destroyed ;[TD001]
2816
2816 NXTGRP: ;[TD001]
2817 CLR R0 ;[TD001]
2818 BIT #OP$KGR,$OPTDN ;[TD001]
2819 BNE 10$ ;[TD001]
2820 MOV #RQB+LD.GRO,R0 ;[TD001]
2821 BIT #PF.XDF,$PFLAG ;[TD001]
2822 BNE 20$ ;[TD001]
2823 MOV #G$LEN,R1 ;[TD001]
2824 MOV #G$NAM,R2 ;[TD001]
2825 MOV #-6,R3 ;[TD001]
2826 BR 30$ ;[TD001]
2827 MOV #G$LEN+MAXID,R1 ;[TD001]
2828 MOV #G$GNAM,R2 ;[TD001]
2829 MOV #G$NML,R3 ;[TD001]
2830 CALL FNDBLK ;[TD001]
2831 BCS 99$ ;[TD001]
2832 BISB #F.FND,FLAGS ;[TD001]
2833 CLC ;[TD001]
2834 CLC ;[TD001]
2835 RETURN ;[TD001]

```

2816	021662										
2817	021662	005000									
2818	021664	032767	001000	000000G							
2819	021672	001002									
2820	021674	012700	000024G								
2821	021700	032767	000000G	000000G	10\$:						
2822	021706	001007									
2823	021710	012701	000016								
2824	021714	012702	000002								
2825	021720	012703	177772								
2826	021724	000406									
2827	021726	012701	000036		20\$:						
2828	021732	012702	000016								
2829	021736	012703	000015								
2830	021742				30\$:						
2831	021746	103404									
2832	021750	152767	000004	157616							
2833	021756	000241									
2834	021760				99\$:						

VDIS - VNP SHOW COMMANDS
NXTCIR - GET NEXT CIRCUIT

MACRO V05.03b Monday 15-Jul-85 12:14^{N 4} Page 46

3371
3372
3373
3374
3375
3376
3377
3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3390
3391
3392
3393
3394
3395
3396
3397
3398
3399
3400
3401

025050
025052
025056 103416
025060 026127 000000G 043340
025066 001771
025070 116002 000003
025074 066702 000000G
025100 011202
025102 026227 000000G 045452
025110 001760
025112 000241
025114
025116

```
.SBTTL NXTCIR - GET NEXT CIRCUIT

:
: +
:
: NXTCIR - GET NEXT
:
: INPUTS:
:   R3 = NUMBER OF SLT'S REMAINING TO SCAN
:   R5 = ADDRESS OF SLT OF LINE
:
: OUTPUTS:
:   R0 = SLT ADDRESS
:   R1 = ADDRESS OF DDM PDV
:   R5 = ADDRESS OF SLT OF NEXT LINE IN SYSTEM
:
: ALL OTHER REGISTERS ARE PRESERVED
:
: -
NXTCIR: SAVRG <R2>
10$: CALL NXTLIN
      BCS 20$
      CMP Z,NAM(R1),#*RKM
      BEQ 10$
      MOVB L,DLC(R0),R2
      ADD $PDVTA,R2
      MOV (R2),R2
      CMP Z,NAM(R2),#*RLAB
      BEQ 10$
      CLC
20$: RESRG <R2>
      RETURN

: GET NEXT SYSTEM LINE TO OPERATE ON
: BR IF END OF SYSTEM LINES
: IS THIS A KNX?
: BR IF YES - NOT A CIRCUIT
: GET DLC PDV INDEX
: INDEX INTO PDV VECTOR TABLE
: GET PDV ADDRESS
: IS THIS A LAPB LINE?
: BR IF YES - NOT A CIRCUIT
: INDICATE SUCCESS
: RESTORE R2
```


VDIS - VNP SHOW COMMANDS MACRO V05.03b Monday 15-Jul-85 12:14^{N 5} Page 52-1
DISLOO - DISPLAY LOOPBACK NODE INFORMATION

```
3914      ; DISPLAY MESSAGES
3915      ;
3916 027774      ERMSG$ <%NLoopback%NNode      Line%N>
3917 030023      ERBLK$ NLMSG1
3918
3919 030026      ERMSG$ <%A      %A>
3920 030033      ERBLK$ NLMSG2,<TEMP1,TEMP3>
3921
3922      .DSABL LC
```

VDIS - VNP SHOW COMMANDS MACRC V05.03b Monday 15-Jul-85 12:14^{N.6} Page 55-3
DISLIN - DISPLAY LINE INFORMATION

4546
4547 034162
4548 034163
4549
4550

ERMSG\$ < >
ERBLK\$ LMSG12
.DSABL LC

VDIS - VNP SHOW COMMANDS MACRO V05.03b Monday 15-Jul-85 12:14 ^{N 7} Page 66
 MSGOUT - OUTPUT MESSAGE TO TI:

4986
 4987
 4988
 4989
 4990
 4991
 4992
 4993
 4994
 4995
 4996
 4997
 4998
 4999
 5000
 5001
 5002
 5003
 5004
 5005
 5006
 5007

035606
 035612 016700 000000G
 035616
 035622
 035626
 000001

.SBTTL MSGOUT - OUTPUT MESSAGE TO TI:

```

;+
; MSGOUT - OUTPUT MESSAGE TO TI:
; INPUTS:
;   R2 = ARGUMENT BLOCK ADDRESS FOR $ERMSG
; OUTPUTS:
;   THE MESSAGE IS OUTPUT TO TI:
;-

MSGOUT: CALL    $SAVAL          ; SAVE ALL REGISTERS
        MOV     $CLBUF,R0      ; GET ADDRESS OF OUTPUT BUFFER
        CALL    $ERMSG        ; FORMAT THE OUTPUT MESSAGE
        CALL    $TIOUT        ; OUTPUT THE MESSAGE
        RETURN

        .END
  
```

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 6 N 8
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
HSD29	000014	42-3175
HSGNAM	000050	34-2548
HSLDTE	000002	35-2634
HNETW	000024	34-2551
HSPVC	000006	27-1948 27-2002 27-2090
HBRDTE	000004	31-2328
HSRNW	000042	33-2429
HXX29C	000040	42-3167
INSPCT	000067 R	#6-130 48-3474
IO.ATT	= ***** GX	6-267
IO.DET	= ***** GX	6-268
K.CSR	000002	55-4473 55-4477
K.PRT	000000	55-4484
K.VCT	000001	55-4479
LACTIV	000441 R	#6-195 28-2169
LCNTRL	000567 R	#6-214 53-3987 55-4412
LD.ALT	= 000030	#5-99
LD.CEX	= 000024	#5-99
LD.CIR	= 000035	#5-99
LD.LIN	= 000025	#5-99
LD.LOG	= 000031	#5-99
LD.MOD	= 000037	#5-99
LD.NOD	= 000027	#5-99
LD.OBJ	= 000032	#5-99
LD.PRO	= 000026	#5-99
LF.ACT	= 100000	#5-96
LF.BRO	= 000400	#5-96 27-1962 27-2103 53-3968 53-3985 53-4108 53-4143 53-4170 55-4415
		55-4503
LF.BWT	= 000007	#5-96
LF.ENA	= 002000	#5-96 27-1966 28-2178 53-4140 55-4507
LF.LPB	= 001000	#5-96 53-4188
LF.MDC	= 000100	#5-96
LF.MFL	= 004000	#5-96
LF.MTP	= 000020	#5-96 53-3990 53-3999 55-4420 57-4654
LF.PAC	= 000203	#5-96
LF.RDY	= 040000	#5-96 53-4012 53-4041 53-4072 53-4135 53-4162 53-4233 55-4500
LF.REA	= 010000	#5-96
LF.SER	= 000040	#5-96 53-4166
LF.TIM	= 000010	#5-96
LF.UNL	= 020000	#5-96
LF.X2P	= 000000	#5-96
LIERR1	015536 R	28-2234 #28-2241
LKNOWN	000456 R	#6-196 28-2216
LLAB	000645 R	#6-218 55-4422
LMSG1	033732 R	55-4398 #55-4530
LMSG12	034164 R	55-4519 #55-4548
LMSG2	033756 R	55-4405 #55-4533
LMSG3	034014 R	55-4454 #55-4536
LMSG4	034106 R	55-4492 #55-4539
LMSG6	034134 R	55-4517 #55-4542
LMSG9	034156 R	55-4456 #55-4545
LN\$OFF	= 000001	35-2645

VDIS CREATED BY MACRO ON 15-JUL-85 AT 12:17 PAGE 19 N 9
 SYMBOL CROSS REFERENCE CREF 04.00

SYMBOL	VALUE	REFERENCES
		42-3182 44-3277 44-3288 44-3294 44-3306 44-3307 45-3365 49-3588 53-4022
		53-4026 53-4029 53-4050 53-4054 53-4081 53-4085 53-4088 53-4209 53-4259
		53-4260 53-4261 53-4265 53-4269 53-4270 54-4343 54-4346 54-4347 54-4354
		55-4464 58-4683 58-4684 59-4722 59-4726 59-4733 59-4737 59-4744 59-4750
		60-4779 *62-4847 62-4848 62-4851 62-4853 63-4872 22-1295 23-1349
\$CBDMG	= ***** GX	19-1132 19-1142 19-1154 21-1245 21-1254 21-1263
\$CBOMG	= ***** GX	15-903 15-907 15-928 16-954 16-954
\$CBTA	= ***** GX	56-4587 56-4597 57-4639 57-4649 57-4662
\$CCBNM	= ***** GX	8-353
\$CEACC	= ***** GX	22-1307 25-1407 37-2766 42-3179 44-3279 45-3363 59-4747
\$CLBUF	= ***** GX	66-5001
\$CSTA	= ***** GX	15-889 15-891 15-912 15-921 21-1236 53-4119 55-4447 56-4576 57-4628
\$DALI	= 002512 RG	#10-455
\$DBG\$	= *****	27-1922 29-2257
\$DCIR	= 014042 RG	#27-1922
\$DECP	= ***** GX	10-461 18-1023 22-1298 25-1397 26-1439 26-1484 26-1524 26-1605 26-1684
		26-1706 26-1713 53-4208 58-4682 59-4743
\$DEVHD	= ***** GX	62-4847
\$DIV	= ***** GX	49-3583
\$DLIN	= 015160 RG	#28-2155
\$DLOG	= 003446 RG	#12-663
\$DMOD	= 015540 RG	#29-2257
\$DNOD	= 007426 RG	#26-1433
\$DOBJ	= 002272 RG	#9-383
\$DPRO	= 003164 RG	#11-571
\$DSYS	= 001706 RG	#8-339
\$ERMSG	= ***** GX	66-5002
\$EROUT	= ***** GX	8-341 9-385 9-425 10-457 10-535 10-538 11-573 11-629 12-666
		12-680 12-684 26-1436 26-1565 27-1925 27-2131 28-2157 28-2234 29-2260
		34-2529 39-2852 42-3171
\$FILHD	= ***** GX	12-724
\$FNAM	= ***** GX	12-678
\$GETAD	= ***** GX	12-731 22-1308 25-1408 34-2570 37-2767 39-2894 42-3180 44-3280 45-3364
		53-4021 53-4025 53-4049 53-4053 53-4080 53-4084 53-4262 59-4749
\$GETRV	= ***** GX	10-461 18-1023 22-1298 25-1397 26-1439 26-1484 26-1524 26-1605 26-1630
		26-1684 26-1706 26-1713 26-1762 26-1763 27-1949 27-2003 27-2091 31-2329
		31-2352 31-2354 33-2430 34-2553 35-2635 37-2741 39-2880 40-2926 42-3168
		49-3577 49-3578 53-4208 53-4258 55-4473 58-4682 59-4733 59-4743 60-4778
		60-4785 62-4850 62-4853
\$HMBUF	= ***** GX	10-460 10-464 18-1022 18-1026 18-1027 18-1030 18-1031 18-1032 18-1033
		22-1297 22-1299 25-1396 25-1398
\$LGCON	= ***** GX	15-894
\$LGFB	= ***** GX	15-898 15-909
\$LGMON	= ***** GX	15-888 15-890
\$LGSTT	= ***** GX	13-832 14-859
\$LGUIT	= ***** GX	15-901 15-905
\$LLCTA	= ***** GX	53-4198
\$MUL	= ***** GX	63-4881 63-4886 63-4897
\$OBJHD	= ***** GX	9-388
\$OPTON	= ***** GX	10-487 10-506 10-519 12-667 *12-669 12-671 *12-704 *12-707 13-822
		13-826 15-883 15-885 17-977 29-2266 30-2305 30-2307 31-2366 32-2404
		33-2433 33-2482 34-2532 34-2537 35-2669 36-2709 37-2778 38-2818 39-2854

VENA - VNP ENABLE/DISABLE LINES MACRO V05.03b Wednesday 11-Sep-85^{N 10} 12:06 Page 5-1
MACRO CALLS

110
111

.ENDC
.ENDM ASR\$

```

617 .SBTTL $VACT - SET ACTIVE POLLING RATE
618
619 : *--$VACT-SET ACTIVE POLLING RATIO FOR A TRIBUTARY
620 :
621 : THIS ROUTINE IS CALLED TO CHANGE THE ACTIVE POLLING RATE
622 : FOR A TRIBUTARY.
623 :
624 : INPUTS:
625 :   $SLTA = SLT ADDRESS
626 :
627 : OUTPUTS:
628 :   C-BIT = SUCCESS/FAILURE
629 :
630 : -
631
632 $VACT::
633 001350 016701 000000G      MOV    $SLTA,R1      : GET SLT ADDRESS
634 001354 032761 000000G 000000G      BIT    #LF,MIP,L.FLG(R1) : IS THIS A MULTIPOINT LINE ?
635 001362 001412          BEQ    101$           : IF EQ, NO - ERROR
636
637 001364          CALL   FNDTRI                : FIND THE DCP TRIBUTARY TABLE
638 001370 103406          BCS    10$             : IF CS, ERROR DETECTED
639 001372 001412          BEQ    102$             : IF EQ, TRIBUTARY TABLE NOT FOUND
640 001374 116767 000022G 000010G      MOVB   RQB+LO.MAC,$BFR+S.PLA : ELSE, SET THE NEW POLLING RATE
641 001402          PUTAD                     : WRITE BACK THE TRIBUTARY TABLE
642 001406          10$·   RETURN
643
644 : ERROR CONDITIONS
645
646 001410          101$:   ERRPT$ ERR31,$EROUT      : LINE IS NOT MULTIPOINT
647 001420          102$:   ERRPT$ ERR33,$EROUT      : INVALID PARAMETER, MULTIPOINT ACTIVE

```

```

1141 003420 001402          BEQ      80$          ; IF EQ, NO
1142 003422 005016          CLR      (SP)         ; ELSE, DON'T CLEAR ACTIVE BIT IN SLT
1143 003424 000404          BR       90$          ; AND LEAVE
1144 003426 062701 000000G  80$:  ADD     #S.LEN,R1      ; GET NEXT SLT TRIB EXTENSION ADDRESS
1145 003432 005302          DEC      R2           ; ANY MORE TRIBUTARIES ?
1146 003434 002366          BGE      70$          ; IF YES - BRANCH
1147 003436 016701 000000G  90$:  MOV     $SLTA,R1      ; ELSE, GET SLT ADDRESS
1148 003442 042661 000000G  DIB      (SP)+,L.FLG(R1) ; CLEAR ACTIVE BIT IF ALL TRIBS ARE 'OFF'
1149 003446          100$:  RETURN
1150
1151          ;
1152          ; ERROR CONDITIONS
1153          ;
1154 003450          101$:  ERRPT$ ERR21,$EROUT

```


1690
 1691
 1692
 1693
 1694
 1695
 1696
 1697
 1698
 1699
 1700
 1701
 1702
 1703
 1704
 1705
 1706 005636
 1707 005636
 1708 005640 020102
 1709 005642 001022
 1710 005644 011246
 1711 005646 010012
 1712 005650
 1713 005660
 1714 005676 012667 000000G
 1715 005702
 1716 005706 000404
 1717
 1718 005710 010046
 1719 005712 010146
 1720 005714
 1721 005720
 1722 005722

.SBTTL QINS - INSERT ENTRY INTO QUEUE

*** - QINS - INSERT ENTRY INTO MIDDLE OF QUEUE

INPUTS:

R0 = UNMAPPED ADDRESS OF CURRENT BLOCK TO BE INSERTED
 R1 = UNMAPPED ADDRESS OF PREVIOUS BLOCK
 R2 = ADDRESS OF LISTHEAD (INTERNAL STORAGE)

OUTPUTS:

R0-R1 = PRESERVED
 R2 = DESTROYED

QINS:

SAVRG <R0> ; SAVE R0
 CMP R1,R2 ; IS PREVIOUS ENTRY THE LISTHEAD ?
 BNE 10\$; IF NE, NO
 MOV (R2),-(SP) ; ELSE, SAVE POINTER TO NEXT ENTRY
 MOV R0,(R2) ; INSERT NEW ENTRY INTO LIST
 CEACCS R0 ; CONVERT TO MAPPED ADDRESS
 GETAD R0,#2 ; READ IN POINTER TO NEXT
 MOV (SP)+,\$BFR ; STORE POINTER TO NEXT
 PUTAD ; WRITE BACK UPDATED LINK
 BR 20\$; AND CONTINUE

10\$: MOV R0,-(SP) ; GET UNMAPPED ADDRESS OF CURRENT BLOCK
 MOV R1,-(SP) ; GET UNMAPPED ADDR OF PREVIOUS BLOCK
 CALL \$XLINK ; LINK CURRENT BLOCK INTO LIST
 20\$: RESRG <R0> ; RESTORE R0
 RETURN

```

2240 010514 152767 000100 000003G BISB #OF,RLU,$BFR+O.FLG ; ELSE, ASSUME LOGIN UIC
2241 010522 032767 000100 000012G BIT #OF,$LOG,RQB+LO.OFL ; RUN UNDER LOGIN UIC ?
2242 010530 001003 BNE 40$ ; IF NE, YES
2243 010532 142767 000100 000003G BICB #OF,RLU,$BFR+O.FLG ; ELSE, RUN UNDER TASK'S UIC
2244 010540 032767 000010 000000G 40$: BIT #OP$LVL,$OPTHN ; HAS THE VERIFICATION STATE CHANGED ?
2245 010546 001411 BEQ 45$ ; IF EQ, NO
2246 010550 042767 177770 000012G BIC #AC<7>,RQB+LO.OFL ; CLEAN OUT JUNK BITS
2247 010556 142767 000007 000003G BICB #7,$BFR+O.FLG ; CLEAN OUT OLD VERIFICATION STATE
2248 010564 156767 000012G 000003G BISB RQB+LO.OFL,$BFR+O.FLG ; STORE NEW VERIFICATION STATE
2249 010572 032767 000004 000000G 45$: BIT #OP$NAM,$OPTHN ; WAS THE NAME SPECIFIED ?
2250 010600 001406 BEQ 47$ ; IF EQ, NO .. WRITE BACK THE OBJECT
2251 010602 016767 000004G 000006G MOV RQB+LO.ONM,$BFR+O.NAM ; ELSE, STORE THE OBJECT NAME
2252 010610 016767 000006G 000010G MOV RQB+LO.ONM+2,$BFR+O.NAM+2 ;
2253 010616 47$: PUTAD ; RE-WRITE OBJECT BLOCK
2254 010622 000456 49$: BR 60$ ; AND RETURN
2255 ;
2256 ; SET NEW OBJECT
2257 ;
2258 010624 012701 000012 50$: MOV #O.LEN,R1 ; GET LENGTH OF OBJECT BLOCK
2259 010630 CALL $ALPOL ; TRY TO ALLOCATE AN OBJECT BLOCK
2260 010634 103456 BCS 102$ ; IF CS, ALLOCATION FAILURE
2261 010636 010067 000164' MOV RO,CURUNM ; SAVE UNMAPPED ADDRESS
2262 010642 012701 000000G MOV #BFR,R1 ; POINT AT I/O BUFFER
2263 010646 012702 000000G MOV #RQB,R2 ; POINT AT REQUEST BLOCK
2264 010652 116261 000002 000002 MOVBL LR,TYP(R2),O.TYP(R1) ; STORE TYPE CODE
2265 010660 116261 000012 000003 MOVBL LO,OFL(R2),O.FLG(R1) ; STORE FLAGS/VERIFICATION LEVEL
2266 010666 116261 000010 000004 MOVBL LO,COP(R2),O.MXC(R1) ; STORE MAXIMUM COPIES VALUE
2267 010674 016261 000004 000006 MOV LO,ONM(R2),O.NAM(R1) ; STORE OBJECT NAME
2268 010702 016261 000006 000010 MOV LO,ONM+2(R2),O.NAM+2(R1) ; ...
2269 ;
2270 ; INSERT ENTRY INTO OBJECT LIST
2271 ;
2272 010710 CEACCS$ R0,R1 ; CONVERT TO MAPPED ADDRESS
2273 010720 55$: PUTAD R1,#O.LEN ; WRITE OUT OBJECT BLOCK
2274 010736 005001 CLR R1 ; GET OBJECT TYPE
2275 010740 156701 000002G BISB $BFR+O.TYP,R1 ; WITHOUT SIGN EXTEND
2276 010744 CALL FINOBJ ; FIND PLACE FOR OBJECT BLOCK
2277 010750 012702 000000G MOV #SOBJHD,R2 ; POINT AT LISTHEAD
2278 010754 CALL QINS ; INSERT ENTRY INTO QUEUE
2279 010760 60$: RETURN
2280 ;
2281 ; ERROR CONDITIONS
2282 ;
2283 010762 101$: ERRPT$ ERR20,$EROUT ; OBJECT NOT IN SYSTEM
2284 010772 102$: ERRPT$ ERR18,$EROUT ; OBJECT BLOCK ALLOCATION FAILURE
2285 011002 103$: ERRPT$ ERR30,$EROUT ; OBJECT MAY NOT BE CLEARED !

```

```

2784 .SBTTL CKFIL - CHECK FILE SINK NAME
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840

;
; CKFIL - CHECK FILE SINK NAME FOR VALIDITY
;
; INPUTS:
;   RQB+LO.SNM - FILE SINK NAME ASCII STRING
;
; OUTPUTS:
;   CARRY - SUCCESS/FAILURE
;
;
; CKFIL:
;
;   MOV R5, -(SP) ; SAVE R5
;   MOV #MXFIL, R1 ; GET MAXIMUM LENGTH OF FILE NAME
;   CALL GETLEN ; GET LENGTH OF FILE NAME STRING
;   BCC 10$ ; BR IF SUCCESSFUL
;   JMP 130$ ; BR IF BAD FILE NAME STRING
;   MOV R0, R5 ; POINT TO END OF STRING
;   MOV #DEFIL, R0 ; GET DEFAULT FILE NAME STRING
;   MOV #FILNAM, R1 ; POINT TO FILE NAMES STORAGE
;   MOV #FILLN, R2 ; GET LENGTH OF DEFAULT FILE NAME
;   MOV (R0)+, (R1)+ ; STORE DEFAULT FILE NAME
;   DEC R2 ; MORE TO TRANSFER?
;   BGT 20$ ; BR IF YES
;
;   ; CHECK DEVICE AND UNIT NUMBER
;
;   MOV R4, R3 ; SAVE CHARACTER COUNT
;   MOV R5, R0 ; GET ADDRESS OF END OF STRING
;   SUB R4, R0 ; POINT TO START OF STRING
;   MOV #FILDEV, R4 ; POINT TO STORAGE FOR FILE NAME
;   CALL CKMDEV ; CHECK FOR VALID DEVICE NAME
;   BCC 40$ ; BR IF FOUND ONE
;   MOV DEFDEV, FILDEV ; STORE DEFAULT DEVICE NAME
;   MOV DEFUNT, FILUNT ; STORE DEFAULT UNIT NUMBER
;   MOV #LO.SNM+RQB, R0 ; POINT TO BEGINNING OF STRING
;   MOV #1, (R0)+ ; ANY DEVICE SPECIFIED?
;   BEQ 120$ ; BR IF YES - BAD DEVICE NAME
;   DEC R3 ; MORE CHARACTERS TO CHECK
;   BGT 30$ ; BR IF YES
;   MOV FILDEV, R0 ; GET DEVICE NAME TO LOOK FOR
;   CALL FNDDCB ; CHECK TO SEE IF DEVICE IS IN SYSTEM
;   BCS 130$ ; BR IF DEVICE NOT IN SYSTEM
;   MOV #LO.SNM+RQB, R0 ; POINT TO BEGINNING OF STRING
;   MOV DUCB+$BFR, R2 ; GET UCB ADDRESS
;   MOV R2, #U.CNT+2 ; READ IN UCB
;   GETRV 42$ ; HAS THIS DEVICE BEEN REDIRECTED?
;   CMP U.RED+$BFR, R2 ; BR IF NO
;   BEQ 45$ ;
;   MOV U.RED+$BFR, R2 ; GET NEXT UCB IN REDIRECT LIST
;   BR 42$ ; CHECK REDIRECT LIST
;   BIT #DV.F11, U.CW1+$BFR ; IS IT A FILES-11 DEVICE?
;   BEQ 120$ ; BR IF NO
;
;   ; CHECK UIC
;

```

